VisiLogic™

SOFTWARE MANUAL

VISILOGIC FUNCTION BLOCKS

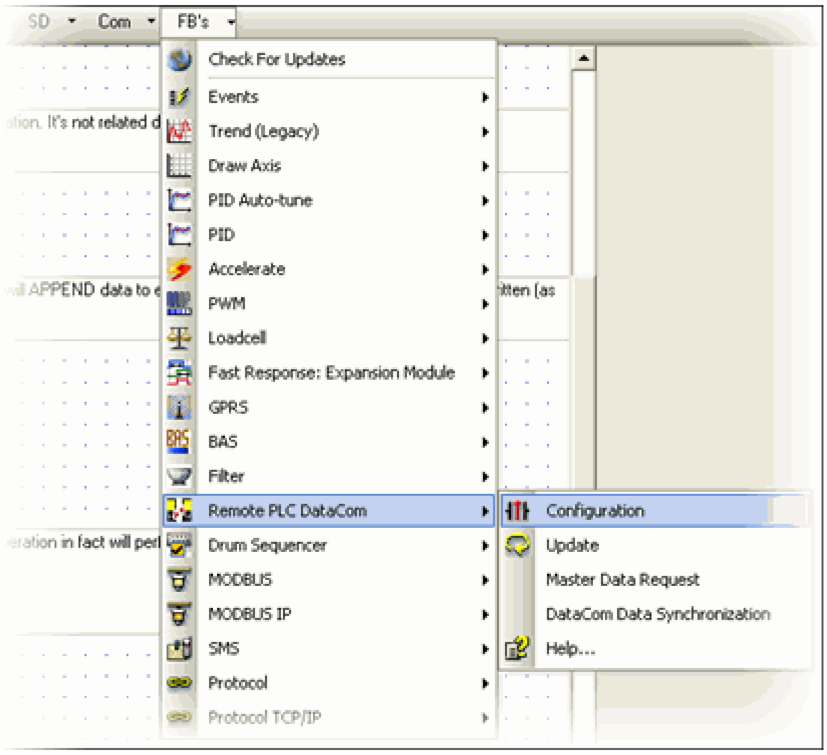
V230-21-G23 Rev: 3:00

UNITRONICS®

Table of Contents

VisiLogic Function Blocks

When you install VisiLogic, the program also installs a Function Block (FB) library for advanced functions, such as SMS messaging and MODBUS communications. FBs that are currently installed in VisiLogic are listed under the FB's menu.

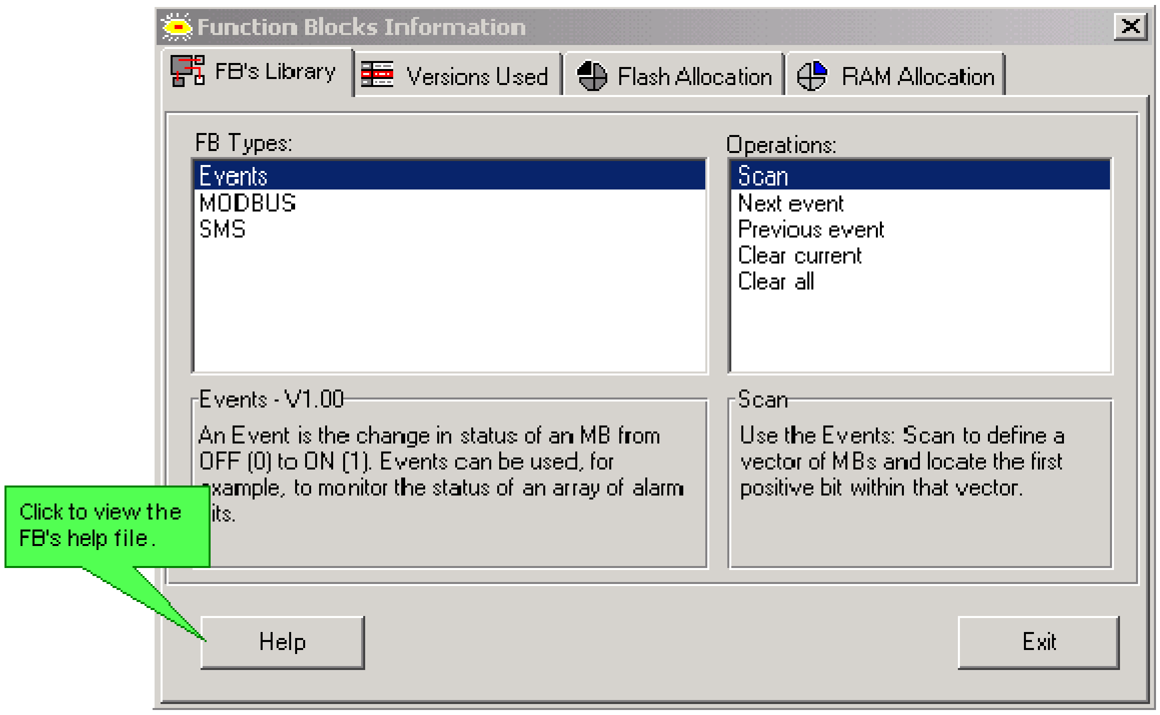


Note • You must use a condition (RLO) to activate any FB that requires Configuration in your application, such as MODBUS or SMS.

Note • To enable Live Update, you can select to use a proxy server in Project Properties.

Use Function Block Information, located on the View menu, to check:

- Which FBs are installed in your library.
- Which FB versions are installed, which versions are used in the open project, and to manage FB versions.
- FB memory usage.



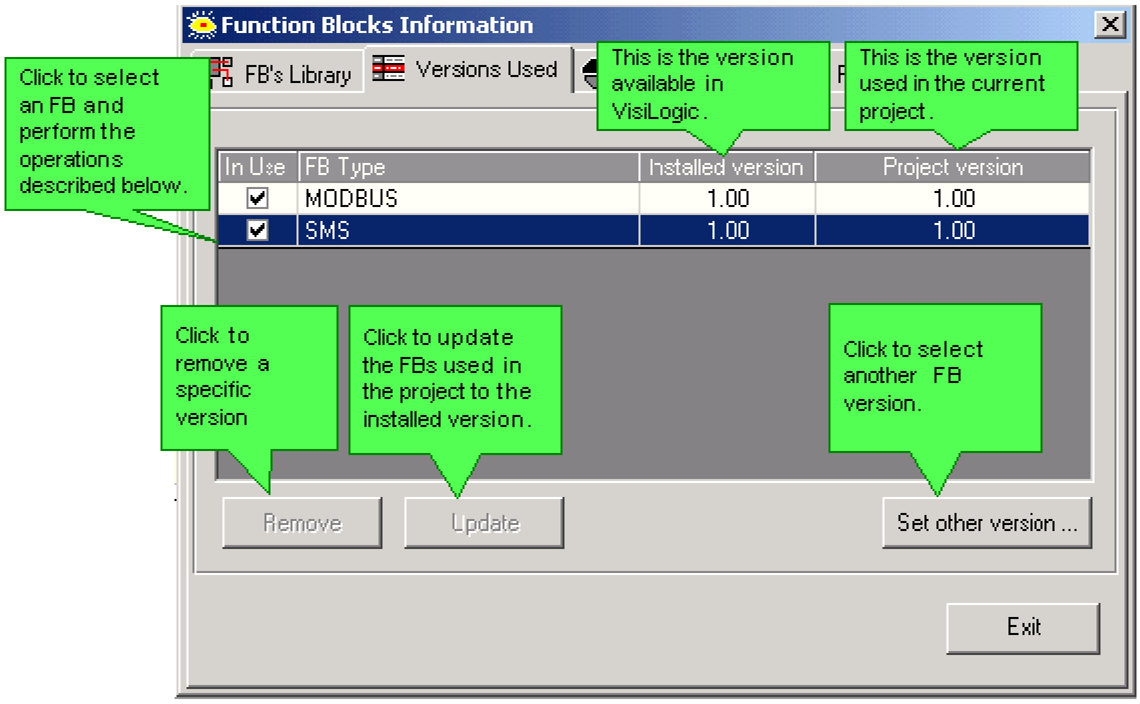
FB Types:
Events
MODBUS
SMS

Operations:
Scan
Next event
Previous event
Clear current
Clear all

Events - V1.00
An Event is the change in status of an MB from OFF (0) to ON (1). Events can be used, for example, to monitor the status of an array of alarm bits.

Scan
Use the Events: Scan to define a vector of MBs and locate the first positive bit within that vector.

Versions Used



Function Blocks Information
FB's Library
Versions Used
Click to select an FB and perform the operations described below.
This is the version available in VisiLogic.
This is the version used in the current project.
Click to remove a specific version
Click to update the FBs used in the project to the installed version.
Click to select another FB version.
Remove
Update
Set other version ...
Exit

In Use	FB Type	Installed version	Project version
✓	MODBUS	1.00	1.00
✓	SMS	1.00	1.00

Updating FB versions

Standard Vision: To install an updated FB library, select Update from the Web from the FBs menu or Help menu, then follow the on-screen instructions. Note that at the end of the download, you must close and then restart VisiLogic. The new FBs will appear on the FBs menu.

Enhanced Vision: FB libraries are updated as part of OS releases. When you update the OS, FBs are automatically updated as well.

FBs List

MODBUS, serial

MODBUS, IP

SMS Messaging

GPRS

Remote PLC DataCom

Communication Protocol

TCP/IP Communication Protocol

PID FB

Drum

Events

MB as PWM

Loadcell

Filter

Accelerate

Fast Response

Draw Axis

BAS

Trends

If your project is configured to Vision controllers that do not support HMI object Trend graphs, the Trend objects will not be displayed in the Project Navigation Window. These controllers include V120/230/260/280/290 (monochrome). In these models, the Trends (Legacy) Function Block may be used.

<u>Examples</u>

Sample applications may be found in the VisiLogic Examples folder, located on the VisiLogic Help menu. This folder contains field-tested VisiLogic (.vlp) sample applications.

MODBUS (Serial)

MODBUS (Serial) Overview

Converting Projects: Vision Divisions

The memory structure of Standard Vision controllers is different from that of Enhanced. Note that if you convert projects, you must make changes according to the information given in the Slave Address tables.

MODBUS enables you to establish master-slave communications between Unitronics PLCs and any connected device that supports the MODBUS protocol. Any controller in the network may function as either master or slave using any of the controller's existing COM Ports.

Within a MODBUS network, you can use standard MODBUS commands to read and write bit and register data; you can also read and write data to Vision controller Data Tables.

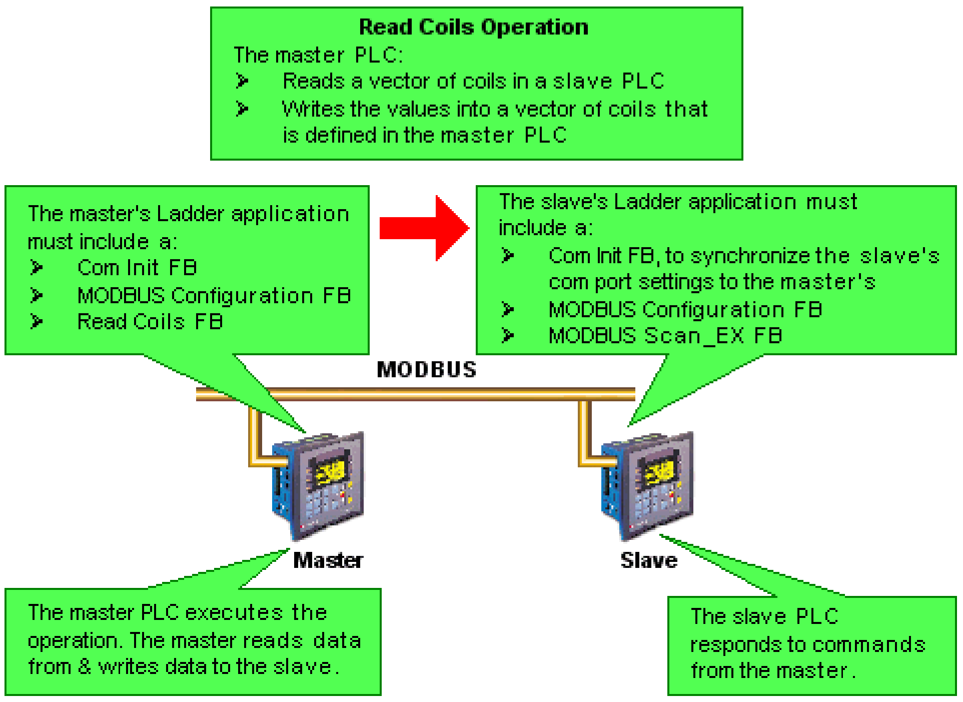
Unitronics currently supports RTU (binary) transmission mode.

Using MODBUS: Unitronics' PLCs, Master - Slave

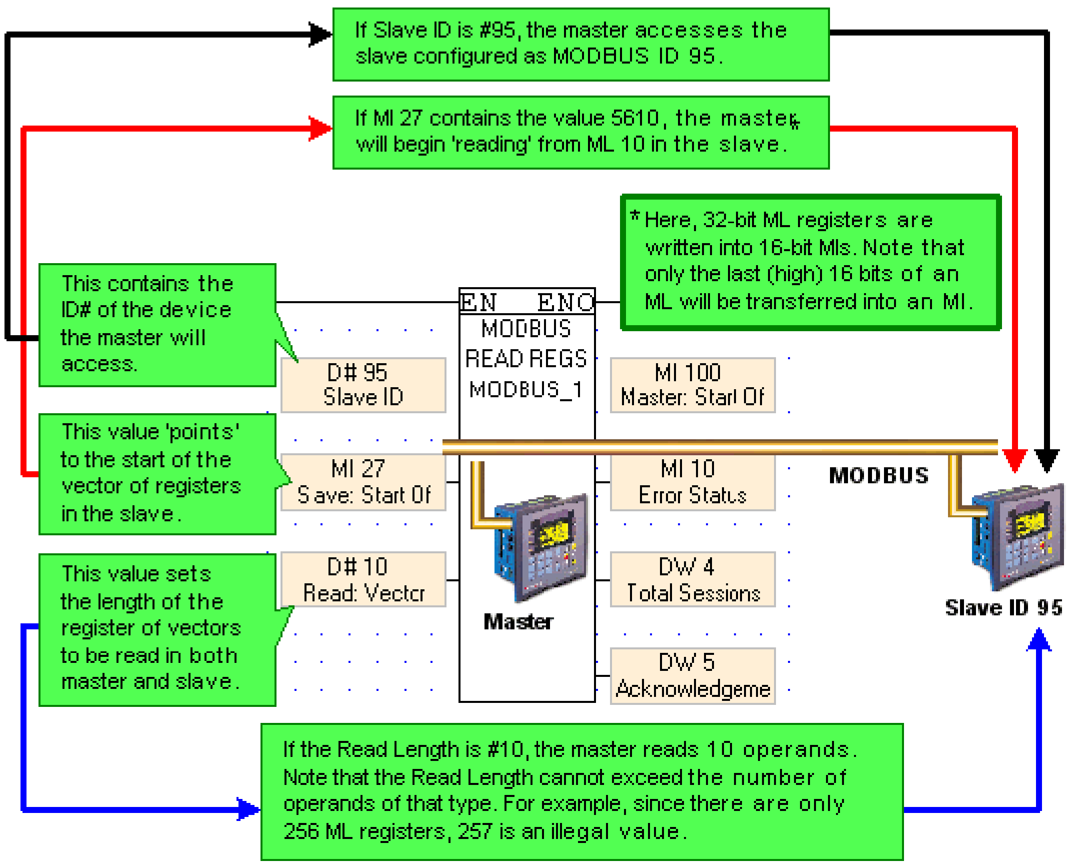
Before using a MODBUS operation in your application, you must:

- Synchronize the communication port settings of master and slave devices. This is done by placing COM Port Init FBs, set with identical parameters, in the ladder application of both master and slave.
- Include at least 1 MODBUS Configuration FB in the ladder application of both master and slave. The port you select must be the same port selected in the COM Port Init FB.
- The condition that activates the Configuration must turn ON for a single program cycle (positive transition recommended). **However, the MODBUS Configuration must be scanned during every program cycle--after the Configuration is activated. One way to ensure this is by placing the Configuration in the first subroutine of the main module.**
- Enable slave devices to be accessed by placing a Scan_EX FB in the slave's Ladder application.

The figure below shows the elements required to carry out a Read Coils Operation.


Read Coils Operation
The master PLC:
➤ Reads a vector of coils in a slave PLC
➤ Writes the values into a vector of coils that is defined in the master PLC
The master's Ladder application must include a:
➤ Com Init FB
➤ MODBUS Configuration FB
➤ Read Coils FB
The slave's Ladder application must include a:
➤ Com Init FB, to synchronize the slave's com port settings to the master's
➤ MODBUS Configuration FB
➤ MODBUS Scan_EX FB
MODBUS
Master
Slave
The master PLC executes the operation. The master reads data from & writes data to the slave.
The slave PLC responds to commands from the master.

Note that the operand addresses in slave PLCs are indirect addresses (pointers).


If Slave ID is #95, the master accesses the slave configured as MODBUS ID 95.
If MI 27 contains the value 5610, the master will begin 'reading' from ML 10 in the slave.
* Here, 32-bit ML registers are written into 16-bit MIs. Note that only the last (high) 16 bits of an ML will be transferred into an MI.
This contains the ID# of the device the master will access.
This value 'points' to the start of the vector of registers in the slave.
This value sets the length of the register of vectors to be read in both master and slave.
EN ENO
MODBUS
READ REGS
MODBUS_1
D# 95
Slave ID
MI 27
Save: Start Of
D# 10
Read: Vector
MI 100
Master: Start Of
MI 10
Error Status
DW 4
Total Sessions
DW 5
Acknowledgeme
MODBUS
Master
Slave ID 95
If the Read Length is #10, the master reads 10 operands. Note that the Read Length cannot exceed the number of operands of that type. For example, since there are only 256 ML registers, 257 is an illegal value.

Using MODBUS: Accessing PLC data via SCADA/OPC server

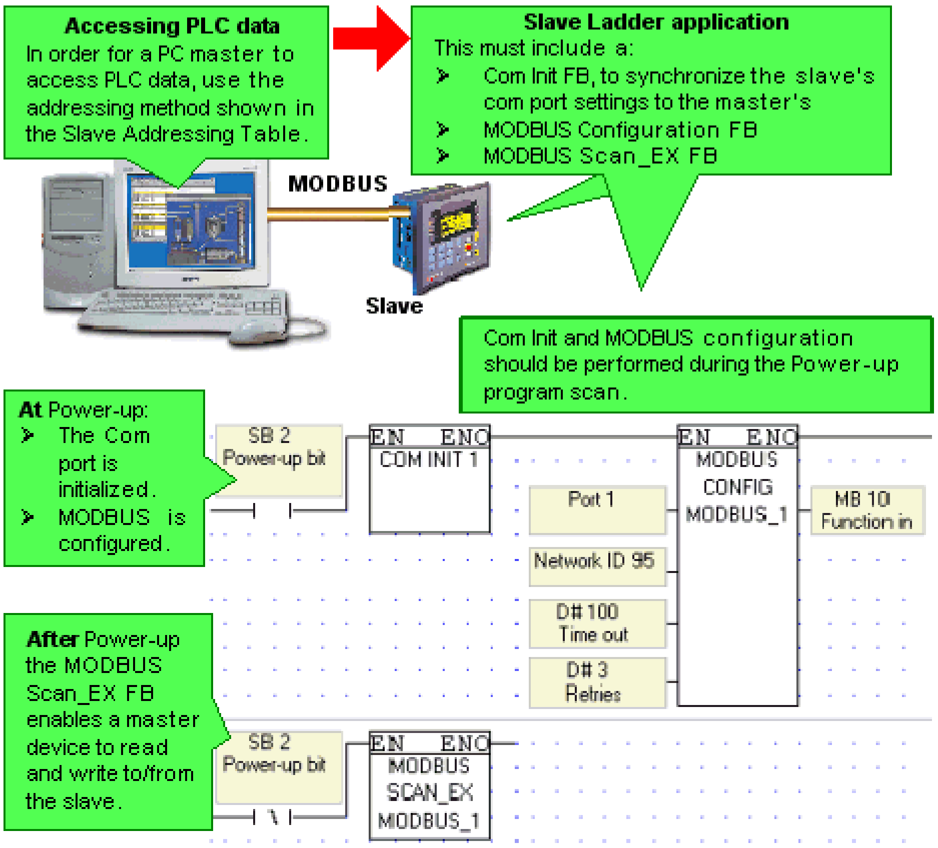
The PC master can access data within the PLC via the addresses given in the Slave Addresses Table.

The PLC slave's Ladder application must include the following:

- A COM Port Init FB.

- A MODBUS Configuration FB. Within the Configuration, the port you select must be the same port selected in the COM Port Init FB.
  Note • The condition that activates the Configuration must turn ON for a single program cycle (positive transition recommended). **However, the MODBUS configuration must be scanned during every program cycle--after the Configuration is activated. One way to ensure this is by placing the configuration in the first subroutine of the main module.**

- A Scan_EX FB


Accessing PLC data
In order for a PC master to access PLC data, use the addressing method shown in the Slave Addressing Table.
Slave Ladder application
This must include a:
➢ Com Init FB, to synchronize the slave's com port settings to the master's
➢ MODBUS Configuration FB
➢ MODBUS Scan_EX FB
MODBUS
Slave
Com Init and MODBUS configuration should be performed during the Power-up program scan.
At Power-up:
➢ The Com port is initialized.
➢ MODBUS is configured.
After Power-up the MODBUS Scan_EX FB enables a master device to read and write to/from the slave.
SB 2 Power-up bit
EN ENO
COM INIT 1
Port 1
Network ID 95
D# 100 Time out
D# 3 Retries
MODBUS CONFIG MODBUS_1
MB 10 Function in
SB 2 Power-up bit
MODBUS SCAN_EX MODBUS_1

Note • The operand addresses in slave PLCs are indirect addresses (pointers).

Note that it is possible to broadcast to the MODBUS network by writing to Slave ID # 0. To do this indirectly addressing the Slave ID to a register, and write 0 to that register.

Slaves: Consecutive References

Whether the MODBUS master is a Unitronics PLC or another device, if the master application size and system requirements allow, it is recommended to add a delay between consecutive references to slaves according to the table below.

Delay (msec) between consecutive references to slaves;
Minimum Baud rate = 9600

	Minimum	Recommended
Jazz	20	40
M90/M91	15	30
V120, V2xx	10	20
V130	5	10
V350, V570	0	5

FB Operations

Operations are grouped under MODBUS on the FB's menu.

MODBUS: Configuration

MODBUS: Scan

MODBUS: Read Coils (1)

MODBUS: Read Inputs (2)

Read Holding Registers (3)

Read Float Registers (3)

Read Input Registers (4)

Read Float Input Registers (4)

Force Coil (5)

Preset Holding Register (6)

Loopback Test (8)

Force Coils (15)

Preset Holding Registers (16)

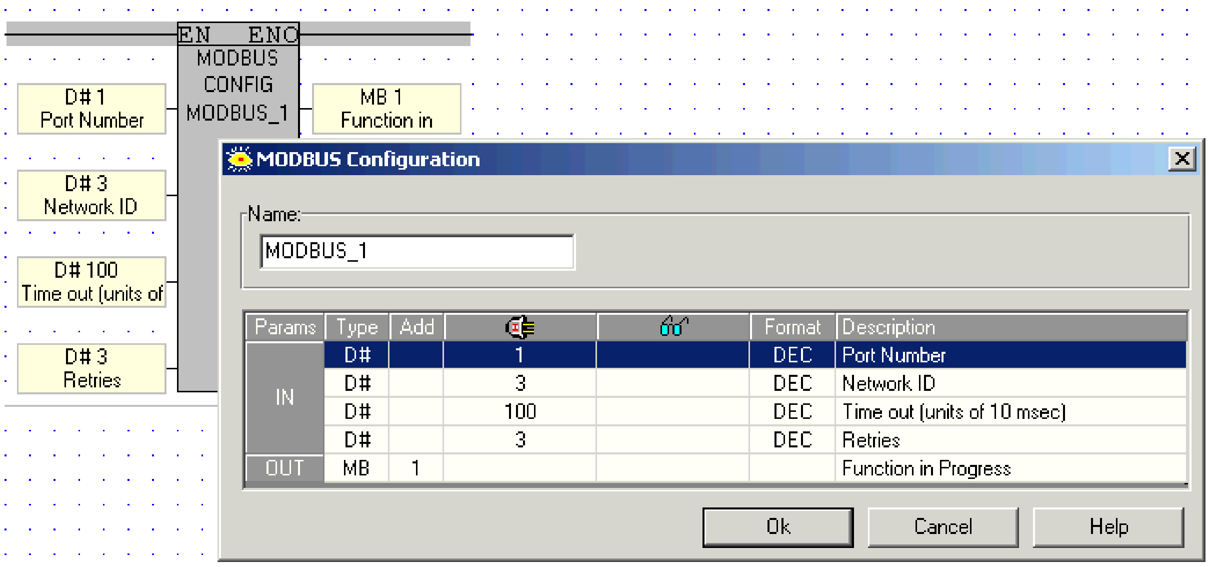
Preset Float Registers (16)

Read/Write Mixed Data

Read/Write to Data Tables

MODBUS: Configuration

A MODBUS Configuration FB must be included in both master and slave Ladder applications as shown below.



Name: MODBUS_1

Params	Type	Add			Format	Description
IN	D#		1		DEC	Port Number
	D#		3		DEC	Network ID
	D#		100		DEC	Time out (units of 10 msec)
	D#		3		DEC	Retries
OUT	MB	1				Function in Progress

Parameter	Type	Function
Port Number	Constant	Click the drop-down arrows to view available ports; click the port you want to use.
Network ID	Constant	This number identifies the device on the network. You can either assign an ID via an MI, or directly via a constant number. The unit ID range is from 0-255. Do not assign the same ID number to more than one device.
Time out	Constant or MI	This is the amount of time a master device will wait for an answer from a slave. Time out units are defined in 10 msecs; a Time out value of 100 is equal to 1 second.
Retries	Constant or MI	This is the number of times a device will try to send a message.
Function in Progress	MB	This bit is ON when MODBUS is active. Use this as a condition bit for MODBUS operations to avoid communication conflicts.

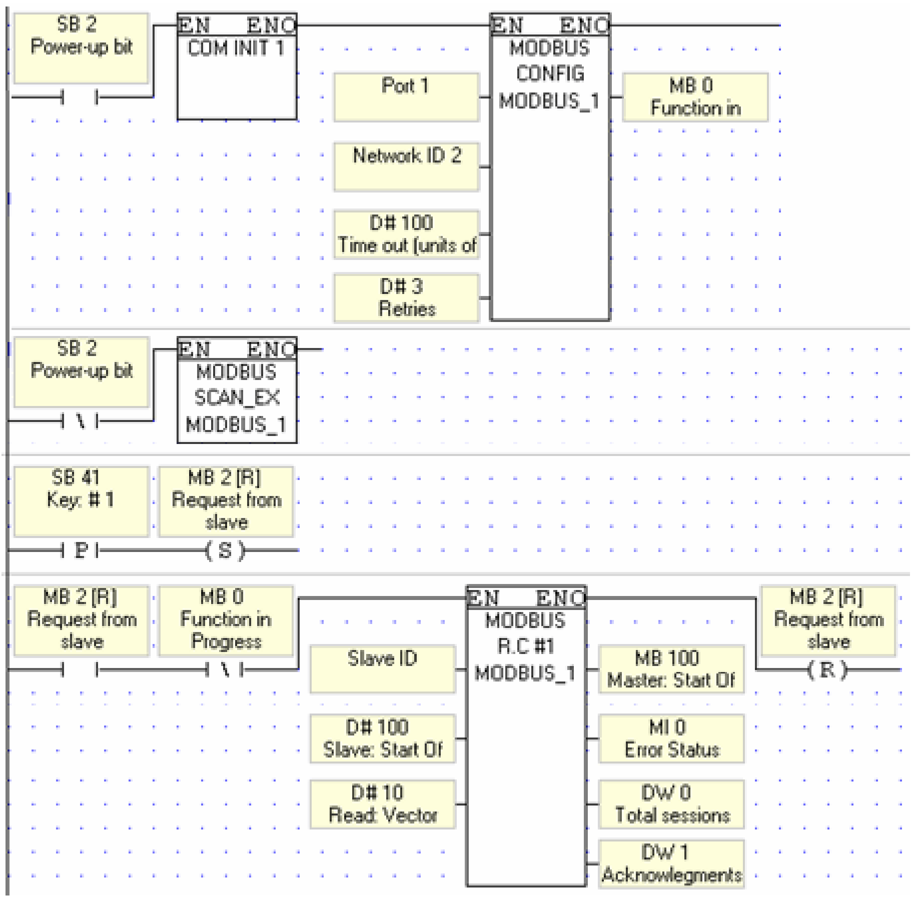
Note • Indirectly addressed parameters in a MODBUS Configuration FB are only read when the Configuration is called. Since a Configuration is generally called as a power-up task, if, for example Retries has been indirectly addressed, and the linked MI is updated, the new value will **not** be read into the Configuration. The value will only be updated until the Configuration is called.

• While a master attempts to send a command, the Function In Progress bit is ON. The number of attempts that the master will make is the number in Retries +1, where '1' is the initial access attempt.

• When a master attempts to access a slave device, and the slave does not answer, the Function In Progress bit will turn ON. This bit will

remain on according to the following:
(the number of retries + 1) x (Time Out), where '1' is the initial access attempt. Note that the Time Out parameter is in units of 10 msec.

The Ladder application below enables the controller act as a MODBUS master and read coils in a slave PLC. The Scan_EX operation shown below enables the controller to also act as a slave.

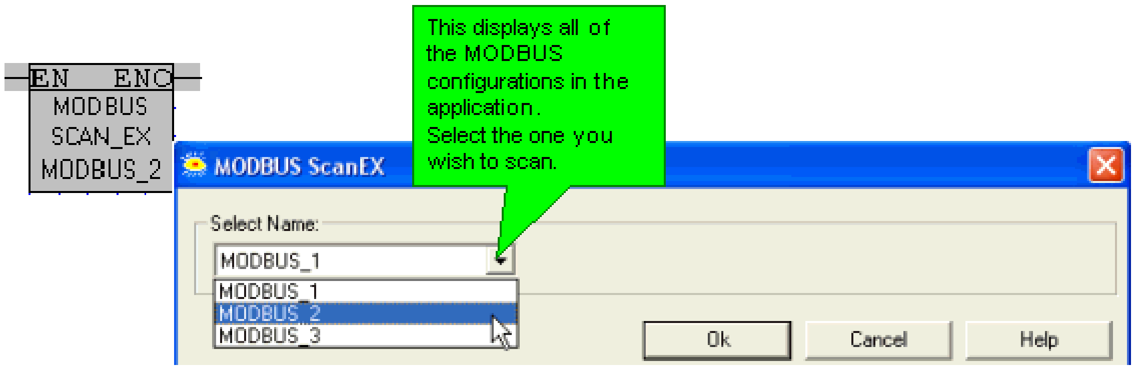


Status Operands

When you place MODBUS operations in your application (Force, Read, Preset, and Loopback commands), you link operands that show the status of MODBUS sessions. Use these to troubleshoot problems.

MODBUS: ScanEX and Scan

Scan_EX enables a master device to access a slave PLC. A Scan_EX must be included in the slave application.


EN ENO
MODBUS
SCAN_EX
MODBUS_2
This displays all of the MODBUS configurations in the application. Select the one you wish to scan.
MODBUS ScanEX
Select Name:
MODBUS_1
MODBUS_1
MODBUS_2
MODBUS_3
Ok
Cancel
Help

Note • Scan_Ex is recommended for new applications.

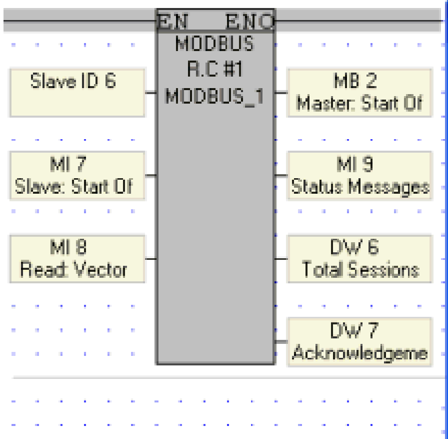
About Scan and Scan_EX

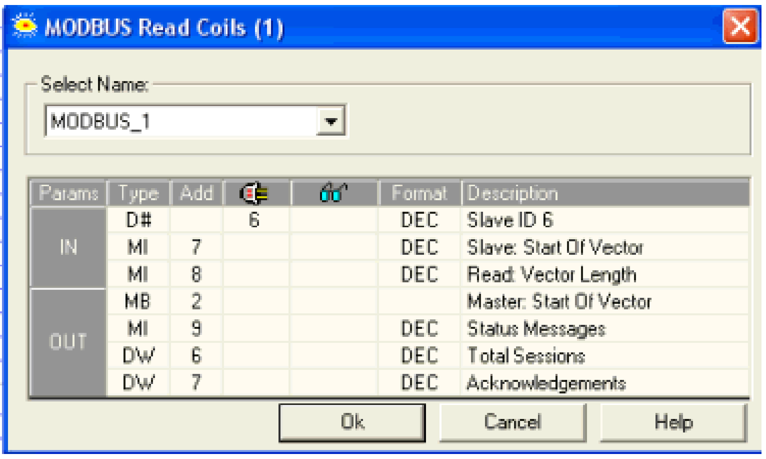
MODBUS Versions **previous to V2.01** offered only the **Scan** FB. Scan is still supported for older, working applications. When MODBUS operations accessed double registers (5100 addresses and higher), using odd addresses, such as 5101, there were incompatibility issues.

When ScanEX receives an input parameter in the 32-bit range (for example, 5100{ML}), it automatically takes double-register values.
If, for example, ScanEX receives a Read Register (6) request for **5100**, it returns the values in 5100 **and** 5101. If, however, ScanEX receives Read Register (6) request for **5101**, it returns Status Message #2-- since 5101 provides the 'high' bytes of the 32-bit register, it is not a legal address.

Read Coils (1)

Use this command to read the status of a selected group of coils and write them into a vector. The coil's status is written into a vector of MBs in the master PLC.




MODBUS Read Coils (1)

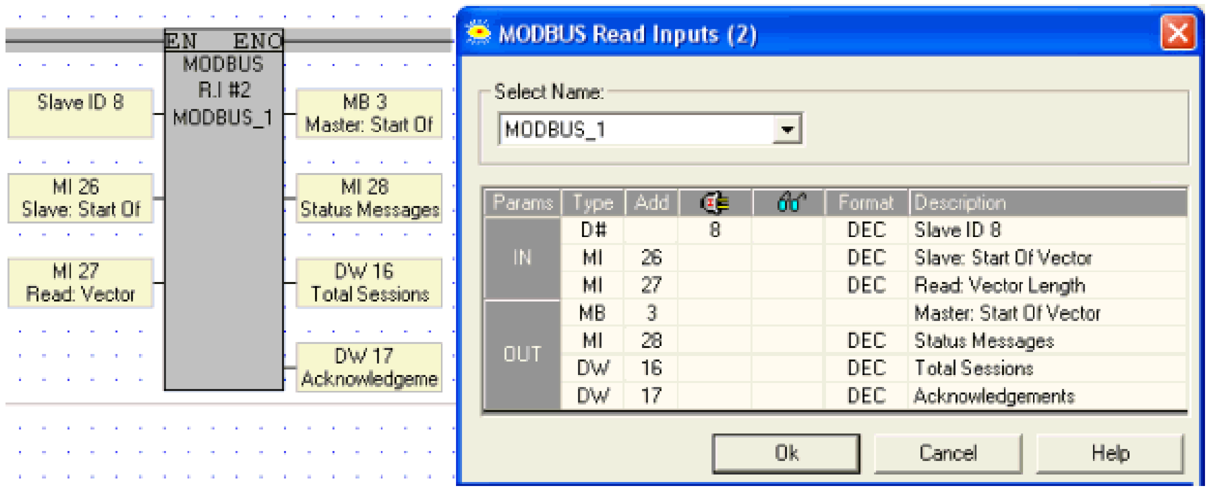
Select Name: MODBUS_1

Params	Type	Add			Format	Description
IN	D#		6		DEC	Slave ID 6
	MI	7			DEC	Slave: Start Of Vector
	MI	8			DEC	Read: Vector Length
OUT	MB	2				Master: Start Of Vector
	MI	9			DEC	Status Messages
	DW	6			DEC	Total Sessions
	DW	7			DEC	Acknowledgements

Parameter	Type	Function
Slave ID	Constant or MI	The ID of the slave device containing the coils to be read (data source).
Slave: Start of Vector	Const, MI, ML, DW	The start of the vector of coils to be read (data source). **Note** ● Check topic Slave Address Tables
Read: Vector Length	Constant or MI	The vector length. **Note** ● A MODBUS command cannot read/write more than 1900 bit operands at one time. In addition, 0 is not a legal length.
Master: Start of Vector	MB	This is the start of a vector of MBs that will contain the coils' status in the master (data destination).
Status Messages	MI	Shows a message number. To check status and diagnose errors, check the MODBUS Status Messages.
Total Sessions	DW	This is the number of times the master PLC will attempt to access the slave device. Note that this is a simple incremental counter. Initialize it by storing 0 into the selected DW.
Acknowledgements	DW	This is the number of times the slave device answers.

Read Inputs (2)

Use this command to read the status of a selected group of inputs in a slave device and write them into a vector. The inputs' status is written into a vector of MBs in the master PLC.

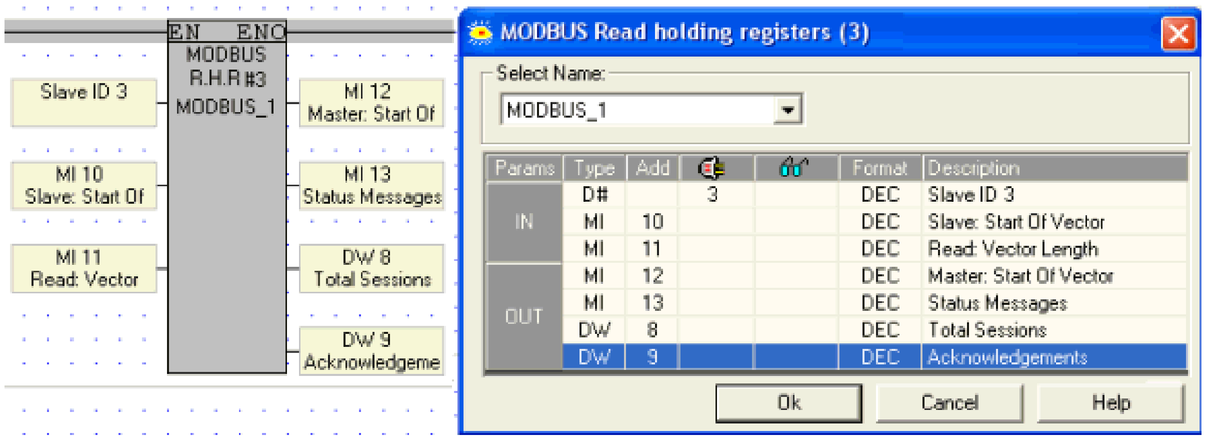


Params	Type	Add			Format	Description
IN	D#		8		DEC	Slave ID 8
	MI	26			DEC	Slave: Start Of Vector
	MI	27			DEC	Read: Vector Length
OUT	MB	3				Master: Start Of Vector
	MI	28			DEC	Status Messages
	DW	16			DEC	Total Sessions
	DW	17			DEC	Acknowledgements

Parameter	Type	Function
Slave ID	Constant or MI	The ID of the slave device containing the inputs to be read (data source).
Slave: Start of Vector	Const, MI, ML, DW	The start of the vector of inputs to be read (data source). **Note •** Check topic Slave Address Tables
Read: Vector Length	Constant or MI	The vector length. **Note •** A MODBUS command cannot read/write more than 1900 bit operands at one time. In addition, 0 is not a legal length.
Master: Start of Vector	MB	This is the start of a vector of MBs that will contain the inputs' status in the master (data destination).
Status Messages	MI	Shows a message number. To check status and diagnose errors, check the MODBUS Status Messages.
Total Sessions	DW	This is the number of times the master PLC will attempt to access the slave device. Note that this is a simple incremental counter. Initialize it by storing 0 into the selected DW.
Acknowledgements	DW	This is the number of times the slave device answers.

Read Holding Registers (3)

Use this command to read the values of a selected group of registers in a slave PLC and write them into a defined vector of registers in the master.



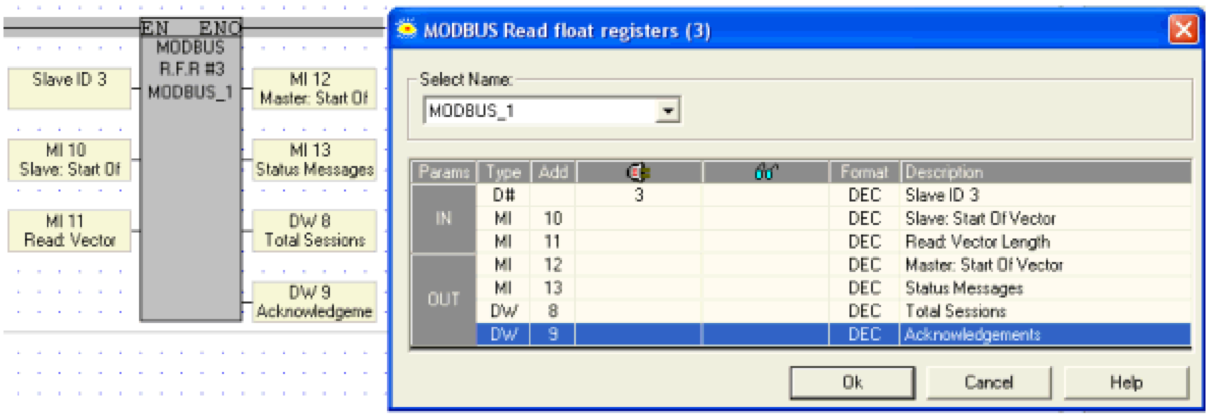
Select Name: MODBUS_1

Params	Type	Add			Format	Description
IN	D#		3		DEC	Slave ID 3
	MI	10			DEC	Slave: Start Of Vector
	MI	11			DEC	Read: Vector Length
OUT	MI	12			DEC	Master: Start Of Vector
	MI	13			DEC	Status Messages
	DW	8			DEC	Total Sessions
	DW	9			DEC	Acknowledgements

Parameter	Type	Function
Slave ID	Constant or MI	The ID of the device containing the registers to be read (data source).
Slave: Start of Vector	Const, MI, ML, DW	The start of the vector of registers to be read (data source). **Note** ● Check topic Slave Address Tables
Read: Vector Length	Const, MI, ML, DW	The vector length **Note** ● A MODBUS command cannot read more than 124 16-bit integers, 62 double registers, or 64 float registers at one time. In addition, 0 is not a legal length. ● If, within the Slave: Start of Vector parameter, the selected register type is a **32-bit double register** (slave addresses 5100 and greater)the preset vector length must be doubled as well. If, for example: - Slave: Start of Vector parameter is set to 6300, and - You wish to preset 4 registers, for a total of 16 bytes - You must set the Preset Vector length to 8. Note that this means that, in these cases, the Preset: Vector Length parameter will always be an even number.
Master: Start of Vector	MI	This is the start of a vector of MIs that will contain the registers' values in the master (data destination).
Status Messages	MI	Shows a message number. To check status and diagnose errors, check the MODBUS Status Messages.
Total Sessions	DW	This is the number of times the master PLC will attempt to access the slave device. Note that this is a simple incremental counter. Initialize it by storing 0 into the selected DW.
Acknowledgements	DW	This is the number of times the slave device answers.

Read Float Registers (3)

Use this command to read the values of a selected group of floating point registers in a slave device and write them into a defined vector of registers in the master. Values after the decimal point are rounded to the nearest whole value.



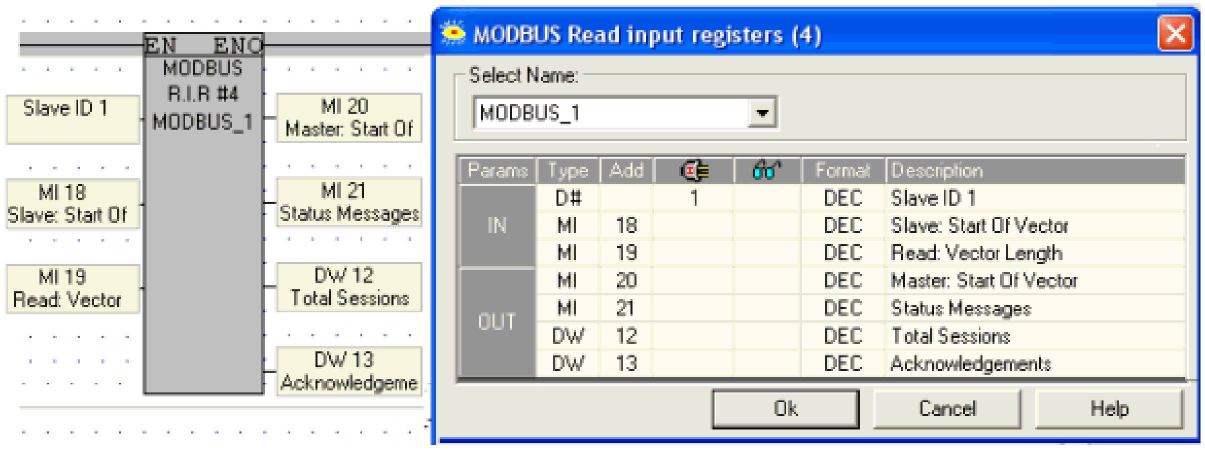
MODBUS_1

Params	Type	Add			Format	Description
IN	D#		3		DEC	Slave ID 3
	MI	10			DEC	Slave: Start Of Vector
	MI	11			DEC	Read: Vector Length
OUT	MI	12			DEC	Master: Start Of Vector
	MI	13			DEC	Status Messages
	DW	8			DEC	Total Sessions
	DW	9			DEC	Acknowledgements

Parameter	Type	Function
Slave ID	Constant or MI	The ID of the device containing the registers to be read (data source).
Slave: Start of Vector	Const, MI, ML, DW	The start of the vector of registers to be read (data source). **Note** ● Check topic Slave Address Tables
Read: Vector Length	Const, MI, ML, DW	The vector length **Note** ● A MODBUS command cannot read more than 124 16-bit integers, 62 double registers, or 64 float registers at one time. In addition, 0 is not a legal length. ● If, within the Slave: Start of Vector parameter, the selected register type is a **32-bit double register** (slave addresses 5100 and greater)the preset vector length must be doubled as well. If, for example: - Slave: Start of Vector parameter is set to 6300, and - You wish to preset 4 registers, for a total of 16 bytes - You must set the Preset Vector length to 8. Note that this means that, in these cases, the Preset: Vector Length parameter will always be an even number. ● You can transpose 16 bits of a 32-bit double register value by turning SB 102 MODBUS Read Long ON in your program. SB 102 is OFF by default, and must be reset by the user program.
Master: Start of Vector	MI	This is the start of a vector of MIs that will contain the registers' values in the master (data destination).
Status Messages	MI	Shows a message number. To check status and diagnose errors, check the MODBUS Status Messages.
Total Sessions	DW	This is the number of times the master PLC will attempt to access the slave device. Note that this is a simple incremental counter. Initialize it by storing 0 into the selected DW.
Acknowledgements	DW	This is the number of times the slave device answers.

Read Input Registers (4)

Use this command to read the values of a selected group of registers in a slave PLC and write them into a defined vector of registers in the master.



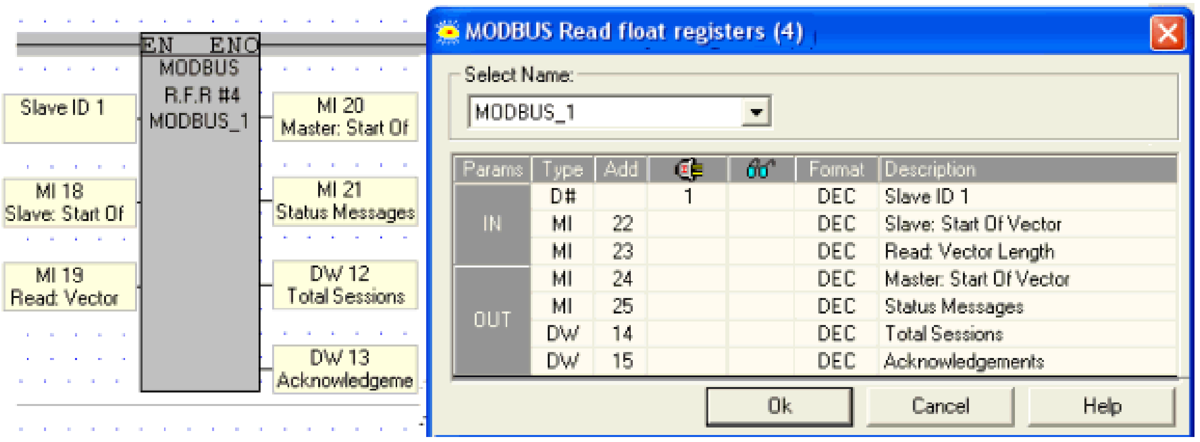
MODBUS_1

Params	Type	Add			Format	Description
IN	D#		1		DEC	Slave ID 1
	MI	18			DEC	Slave: Start Of Vector
	MI	19			DEC	Read: Vector Length
OUT	MI	20			DEC	Master: Start Of Vector
	MI	21			DEC	Status Messages
	DW	12			DEC	Total Sessions
	DW	13			DEC	Acknowledgements

Parameter	Type	Function
Slave ID	Constant or MI	The ID of the device containing the registers to be read (data source).
Slave: Start of Vector	Const, MI, ML, DW	The start of the vector of registers to be read (data source). **Note** ● Check topic Slave Address Tables
Read: Vector Length	Const, MI, ML, DW	The vector length **Note** ● A MODBUS command cannot read more than 124 16-bit integers, 62 double registers, or 64 float registers at one time. In addition, 0 is not a legal length. ● If, within the Slave: Start of Vector parameter, the selected register type is a **32-bit double register** (slave addresses 5100 and greater)the preset vector length must be doubled as well. If, for example: - Slave: Start of Vector parameter is set to 6300, and - You wish to preset 4 registers, for a total of 16 bytes - You must set the Preset Vector length to 8. Note that this means that, in these cases, the Preset: Vector Length parameter will always be an even number.
Master: Start of Vector	MI	This is the start of a vector of MIs that will contain the registers' values in the master (data destination).
Status Messages	MI	Shows a message number. To check status and diagnose errors, check the MODBUS Status Messages.
Total Sessions	DW	This is the number of times the master PLC will attempt to access the slave device. Note that this is a simple incremental counter. Initialize it by storing 0 into the selected DW.
Acknowledgements	DW	This is the number of times the slave device answers.

Read Float Registers (4)

Use this command to read the values of a selected group of floating point registers in a slave device and write them into a defined vector of registers in the master. Values after the decimal point are rounded to the nearest whole value.



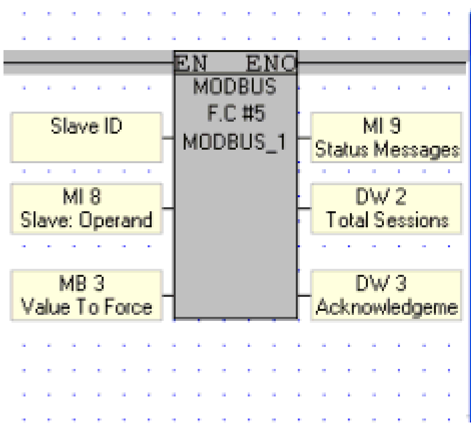
Select Name: MODBUS_1

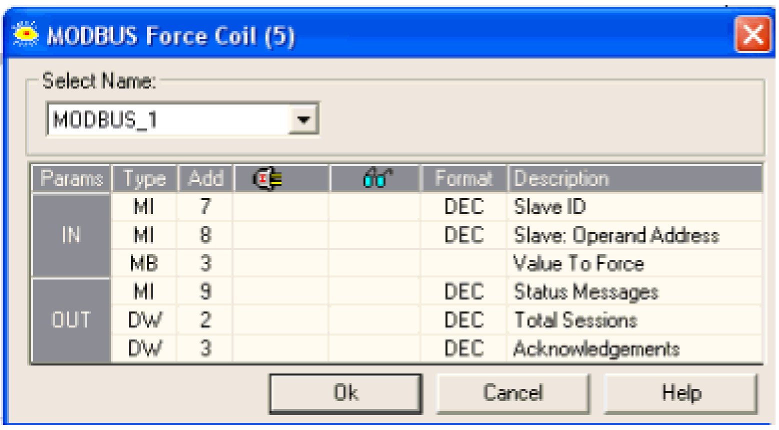
Params	Type	Add			Format	Description
IN	D#		1		DEC	Slave ID 1
	MI	22			DEC	Slave: Start Of Vector
	MI	23			DEC	Read: Vector Length
OUT	MI	24			DEC	Master: Start Of Vector
	MI	25			DEC	Status Messages
	DW	14			DEC	Total Sessions
	DW	15			DEC	Acknowledgements

Parameter	Type	Function
Slave ID	Constant or MI	The ID of the device containing the registers to be read (data source).
Slave: Start of Vector	Const, MI, ML, DW	The start of the vector of registers to be read (data source). **Note** ● Check topic Slave Address Tables
Read: Vector Length	Const, MI, ML, DW	The vector length **Note** ● A MODBUS command cannot read more than 124 16-bit integers, 62 double registers, or 64 float registers at one time. In addition, 0 is not a legal length. ● If, within the Slave: Start of Vector parameter, the selected register type is a **32-bit double register** (slave addresses 5100 and greater)the preset vector length must be doubled as well. If, for example: - Slave: Start of Vector parameter is set to 6300, and - You wish to preset 4 registers, for a total of 16 bytes - You must set the Preset Vector length to 8. Note that this means that, in these cases, the Preset: Vector Length parameter will always be an even number. • You can transpose 16 bits of a 32-bit double register value by turning SB 102 MODBUS Read Long ON in your program. SB 102 is OFF by default, and must be reset by the user program.
Master: Start of Vector	MI	This is the start of a vector of MIs that will contain the registers' values in the master (data destination).
Status Messages	MI	Shows a message number. To check status and diagnose errors, check the MODBUS Status Messages.
Total Sessions	DW	This is the number of times the master PLC will attempt to access the slave device. Note that this is a simple incremental counter. Initialize it by storing 0 into the selected DW.
Acknowledgements	DW	This is the number of times the slave device answers.

Force Coil (5)

Use this command to force the status of a selected coil in a slave PLC. The coil's status is forced according to the status of a selected MB in the master PLC.





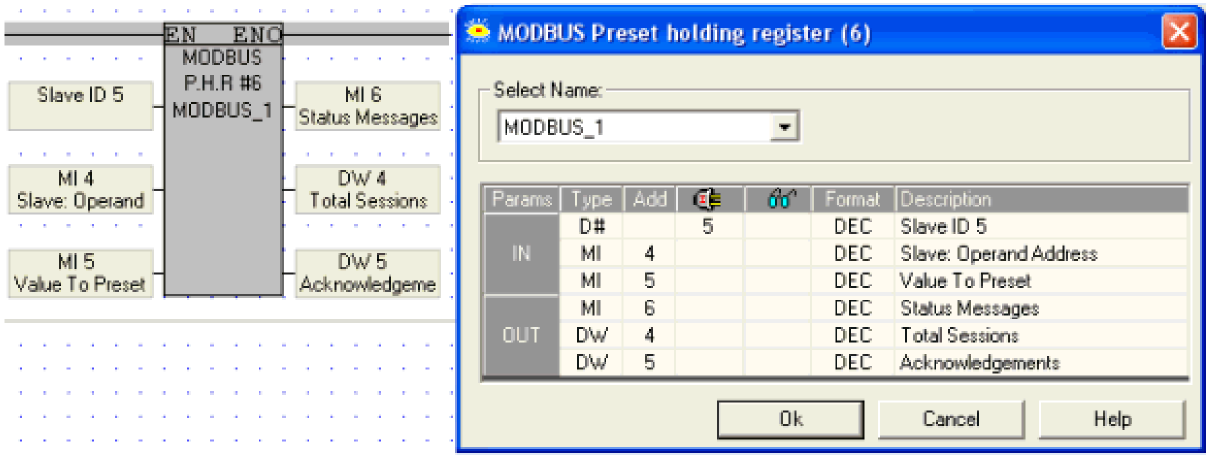
Select Name: MODBUS_1

Params	Type	Add			Format	Description
IN	MI	7			DEC	Slave ID
	MI	8			DEC	Slave: Operand Address
	MB	3				Value To Force
OUT	MI	9			DEC	Status Messages
	DW	2			DEC	Total Sessions
	DW	3			DEC	Acknowledgements

Parameter	Type	Function
Slave ID	Constant or MI	The ID of the device containing the coil to be forced (data source).
Slave Address	Const, MI, ML, DW	The address of the coil to be forced (data target). **Note** ● Check topic Slave Address Tables
Value to Force	M, SB, I, O,T	This MB is located in the master PLC; this MB contains the **status** to be forced (data source). If, for example, the status of this MB is OFF, the status of the coil in the slave will be forced to OFF. **Note** ● A MODBUS command cannot read/write more than 1900 bit operands at one time. In addition, 0 is not a legal length.
Status Messages	MI	Shows a message number. To check status and diagnose errors, check the MODBUS Status Messages.
Total Sessions	DW	This is the number of times the master PLC will attempt to access the slave device. Note that this is a simple incremental counter. Initialize it by storing 0 into the selected DW.
Acknowledgements	DW	This is the number of times the slave device answers.

Preset Holding Register (6)

Use this command to preset the value of a single register in a slave PLC. The value is set in a register contained in the master PLC.

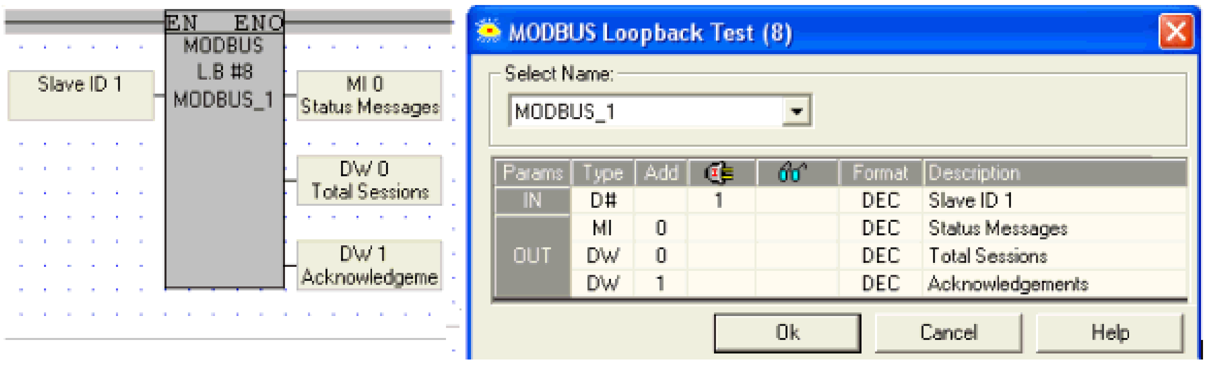


Params	Type	Add			Format	Description
IN	D#		5		DEC	Slave ID 5
	MI	4			DEC	Slave: Operand Address
	MI	5			DEC	Value To Preset
OUT	MI	6			DEC	Status Messages
	DW	4			DEC	Total Sessions
	DW	5			DEC	Acknowledgements

Parameter	**Type**	**Function**
Slave ID	Constant or MI	The ID of the device containing the register to be preset (target).
Slave: Operand Address	Const, MI, ML, DW	The address of the register to be preset (target). **Note** ● Check topic Slave Address Tables
Value to Preset	Constant, MI, SI, ML, SL, DW, SDW or T	This is the address of the register containing the value in the master PLC (source). This value will be written into the slave's register, the register that is to be preset.
Status Messages	MI	Shows a message number. To check status and diagnose errors, check the MODBUS Status Messages.
Total Sessions	DW	This is the number of times the master PLC will attempt to access the slave device. Note that this is a simple incremental counter. Initialize it by storing 0 into the selected DW.
Acknowledgements	DW	This is the number of times the slave device answers.

Loopback Test (8)

Use this command to send a test message to a slave device and receive Acknowledgements when communications are functioning properly.

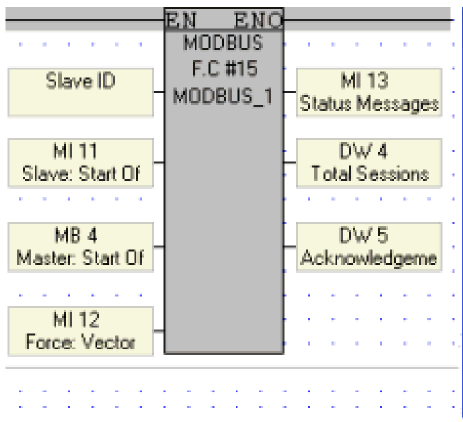


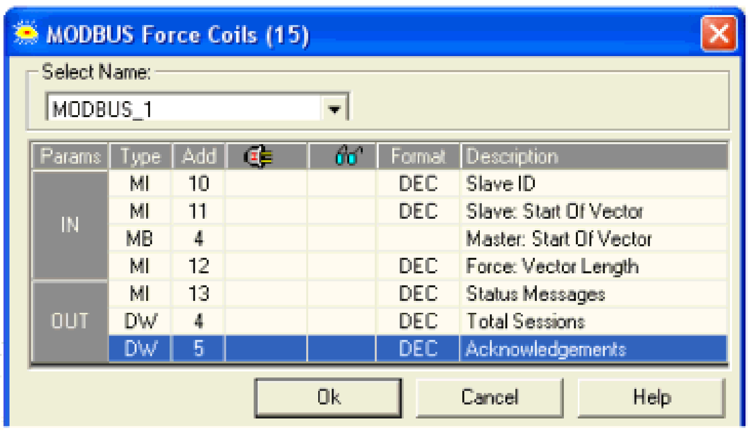
Params	Type	Add			Format	Description
IN	D#		1		DEC	Slave ID 1
OUT	MI	0			DEC	Status Messages
	DW	0			DEC	Total Sessions
	DW	1			DEC	Acknowledgements

Parameter	**Type**	**Function**
Slave ID	Constant or MI	The ID of the device to be checked.
Status Messages	MI	Shows a message number. To check status and diagnose errors, check the MODBUS Status Messages.
Total Sessions	DW	This is the number of times the master PLC will attempt to access the slave device. Note that this is a simple incremental counter. Initialize it by storing 0 into the selected DW.
Acknowledgements	DW	This is the number of times the slave device answers.

Force Coils (15)

Use this command to force the status of a selected group of coils in a slave PLC. The coils' status is forced according to the status of a group of MBs in the master PLC.



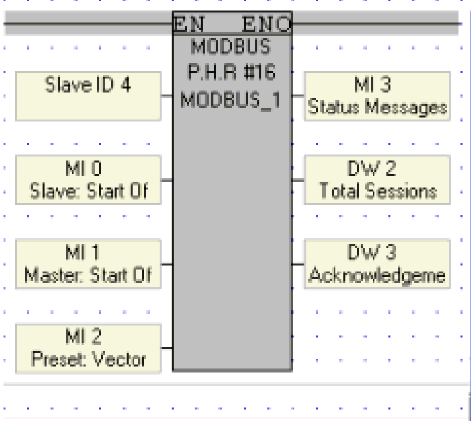


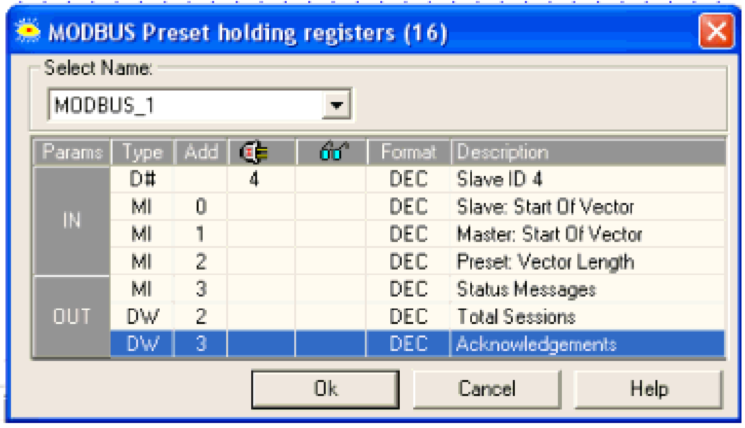
Params	Type	Add			Format	Description
IN	MI	10			DEC	Slave ID
	MI	11			DEC	Slave: Start Of Vector
	MB	4				Master: Start Of Vector
	MI	12			DEC	Force: Vector Length
OUT	MI	13			DEC	Status Messages
	DW	4			DEC	Total Sessions
	DW	5			DEC	Acknowledgements

Parameter	**Type**	**Function**
Slave ID	Constant or MI	The ID of the slave device containing the coils to be forced (target).
Slave:Start of Vector	Const, MI, ML, DW	The start of the vector of coils to be forced (data target). **Note** ● Check topic Slave Address Tables
Master: Start of Vector	MI, SB, I, O,T	This is the start of a vector of MBs that will contain the coils' status in the master (data source).
Force: Vector Length	Constant or MI	The vector length. **Note** ● A MODBUS command cannot read/write more than 1900 bit operands at one time. In addition, 0 is not a legal length.
Status Messages	MI	Shows a message number. To check status and diagnose errors, check the MODBUS Status Messages.
Total Sessions	DW	This is the number of times the master PLC will attempt to access the slave device. Note that this is a simple incremental counter. Initialize it by storing 0 into the selected DW.
Acknowledgements	DW	This is the number of times the slave device answers.

Preset Holding Registers (16)

Use this command to preset the value of a group of registers in a slave PLC. The values are set in a vector of registers contained in the master PLC.



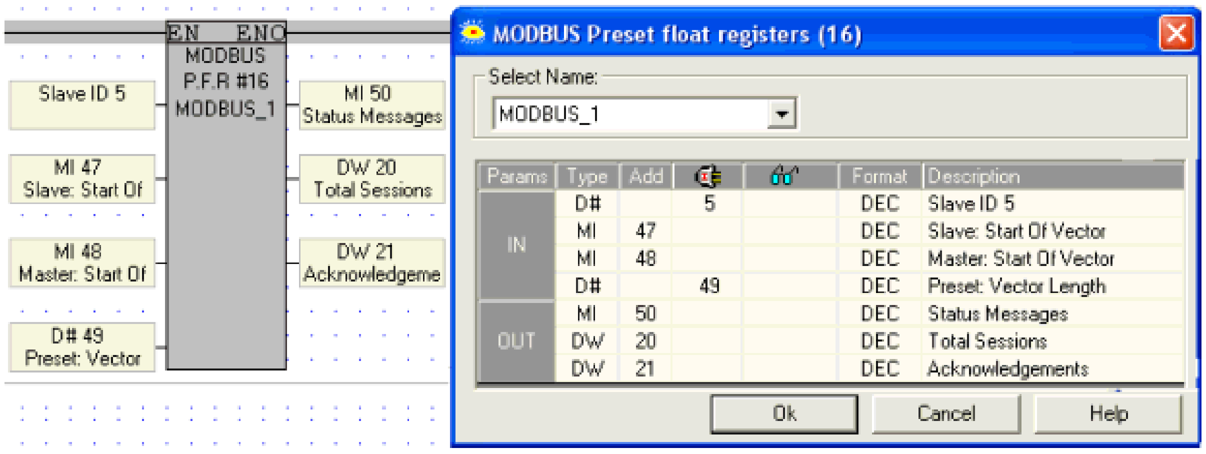


Params	Type	Add			Format	Description
IN	D#		4		DEC	Slave ID 4
	MI	0			DEC	Slave: Start Of Vector
	MI	1			DEC	Master: Start Of Vector
	MI	2			DEC	Preset: Vector Length
OUT	MI	3			DEC	Status Messages
	DW	2			DEC	Total Sessions
	DW	3			DEC	Acknowledgements

Parameter	Type	Function
Slave ID	Constant or MI	The ID of the device containing the registers to be preset (target).
Slave: Start of Vector	Const, MI, ML, DW	The start of the vector of registers to be preset (target). **Note** ● Check topic Slave Address Tables
Master: Start of Vector	Constant, MI, SI, ML, SL, DW, SDW or T	This is the start of a vector of MIs that will contain the registers' values in the master (data source).
Preset: Vector Length	Const, MI, ML, DW	The length of the vector of registers in both master and slave. **Note** ● A MODBUS command cannot read more than 124 16-bit integers, 62 double registers, or 64 float registers at one time. In addition, 0 is not a legal length. ● If, within the Slave: Start of Vector parameter, the selected register type is a **32-bit double register** (slave addresses 5100 and greater)the preset vector length must be doubled as well. If, for example: - Slave: Start of Vector parameter is set to 6300, and - You wish to preset 4 registers, for a total of 16 bytes - You must set the Preset Vector length to 8. Note that this means that, in these cases, the Preset: Vector Length parameter will always be an even number.
Status Messages	MI	Shows a message number. To check status and diagnose errors, check the MODBUS Status Messages.
Total Sessions	DW	This is the number of times the master PLC will attempt to access the slave device. Note that this is a simple incremental counter. Initialize it by storing 0 into the selected DW.
Acknowledgements	DW	This is the number of times the slave device answers.

Preset Float Registers (16)

Use this command to preset the value of a group of floating point registers in a slave PLC. The values are set in a vector of registers contained in the master PLC. Values after the decimal point are rounded to the nearest whole value.



MODBUS Preset float registers (16)

Select Name: MODBUS_1

Params	Type	Add			Format	Description
IN	D#		5		DEC	Slave ID 5
	MI	47			DEC	Slave: Start Of Vector
	MI	48			DEC	Master: Start Of Vector
	D#		49		DEC	Preset: Vector Length
OUT	MI	50			DEC	Status Messages
	DW	20			DEC	Total Sessions
	DW	21			DEC	Acknowledgements

Parameter	Type	Function
Slave ID	Constant or MI	The ID of the device containing the register to be preset (target).
Slave: Start of Vector	Const, MI, ML, DW	The address of the register to be preset (target). **Note** ● Check topic Slave Address Tables
Master: Start of Vector	MI, SI, ML, SL, DW, SDW or T	This is the address of the register containing the value in the master PLC (source). This value will be written into the slave's register, the register that is to be preset.
Preset: Vector Length	Const, MI, ML, DW	The length of the vector of registers in both master and slave. **Note** ● A MODBUS command cannot read more than 124 16-bit integers, 62 double registers, or 64 float registers at one time. In addition, 0 is not a legal length. ● If, within the Slave: Start of Vector parameter, the selected register type is a **32-bit double register** (slave addresses 5100 and greater)the preset vector length must be doubled as well. If, for example: - Slave: Start of Vector parameter is set to 6300, and - You wish to preset 4 registers, for a total of 16 bytes - You must set the Preset Vector length to 8. Note that this means that, in these cases, the Preset: Vector Length parameter will always be an even number.
Status Messages	MI	Shows a message number. To check status and diagnose errors, check the MODBUS Status Messages.
Total Sessions	DW	This is the number of times the master PLC will attempt to access the slave device. Note that this is a simple incremental counter. Initialize it by storing 0 into the selected DW.
Acknowledgements	DW	This is the number of times the slave device answers.

Read/Write from Data Tables

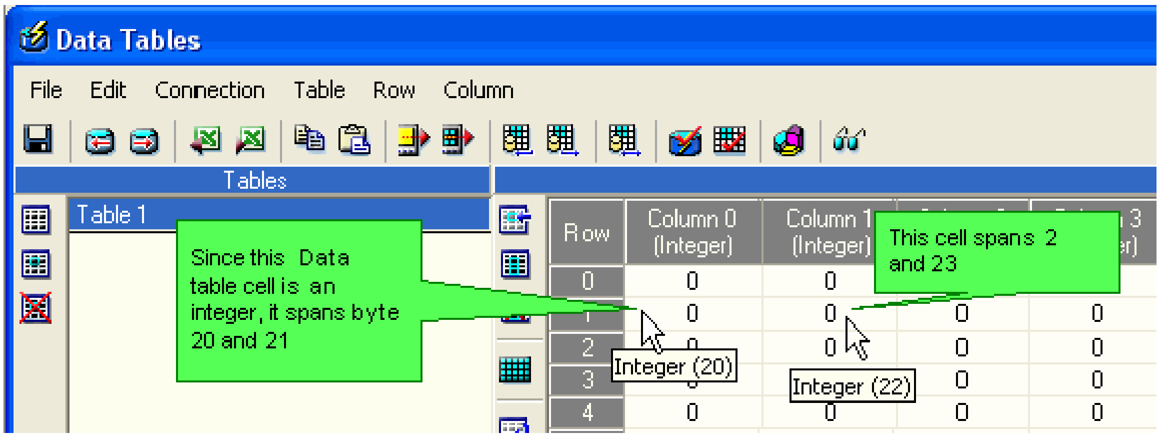
Use these commands to access the bytes in Vision data tables **without** reference to table structure.

Please note that this is not a standard MODBUS function. Read/Write from Data Tables is compatible with Unitronics' Vision PLC data tables only .

To determine the byte number of a data table cell, hold the cursor over the data table cell. A Tooltip opens, displaying the byte number.

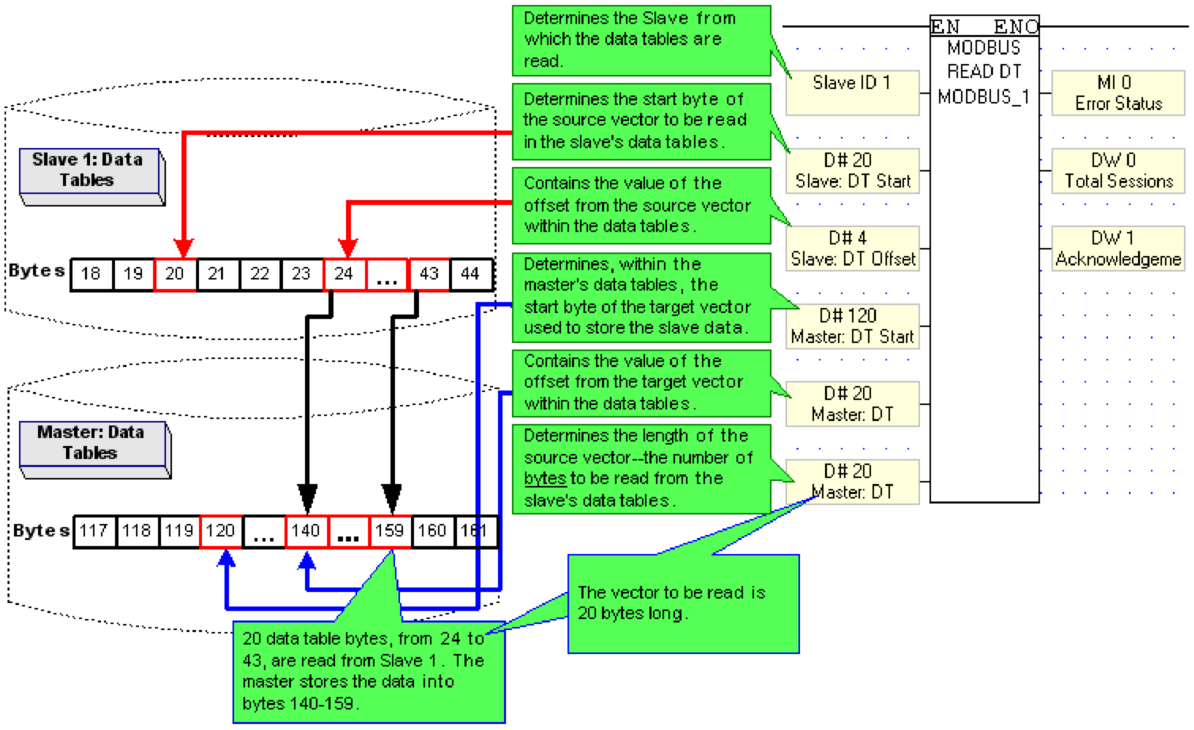
Note • A MODBUS command cannot read/write more than 242 DT bytes at one time.

In addition, 0 is not a legal length.



Read from Data Table

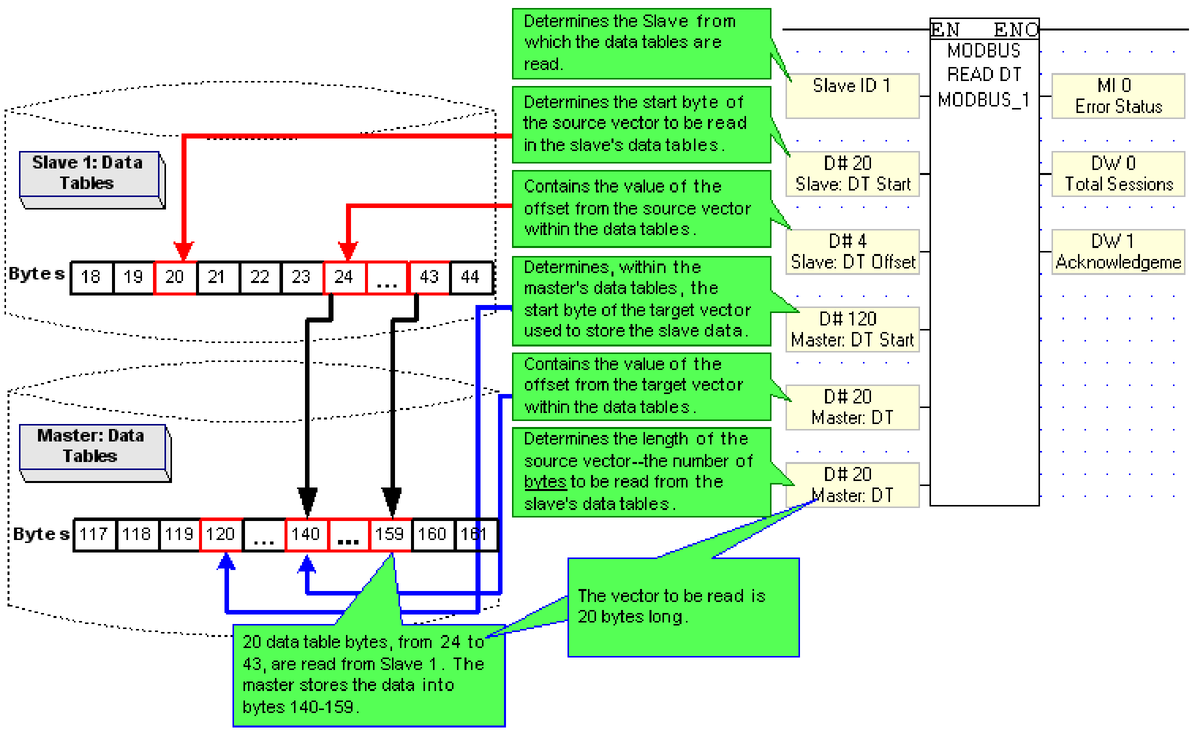
Below, a MODBUS master reads data tables in Slave ID 1. Bytes 24-43 are read from Slave 1 into bytes 140-159 in the master's data tables.


Determines the Slave from which the data tables are read.
Determines the start byte of the source vector to be read in the slave's data tables.
Contains the value of the offset from the source vector within the data tables.
Determines, within the master's data tables, the start byte of the target vector used to store the slave data.
Contains the value of the offset from the target vector within the data tables.
Determines the length of the source vector--the number of bytes to be read from the slave's data tables.
EN ENO
MODBUS READ DT MODBUS_1
Slave ID 1
D# 20 Slave: DT Start
D# 4 Slave: DT Offset
D# 120 Master: DT Start
D# 20 Master: DT
D# 20 Master: DT
MI 0 Error Status
DW 0 Total Sessions
DW 1 Acknowledgeme
Slave 1: Data Tables
Bytes 18 19 20 21 22 23 24 ... 43 44
Master: Data Tables
Bytes 117 118 119 120 ... 140 ... 159 160 161
The vector to be read is 20 bytes long.
20 data table bytes, from 24 to 43, are read from Slave 1. The master stores the data into bytes 140-159.

Parameter	Type	Function
Slave ID	Constant or MI	The ID of the slave device containing the coils to be read (data source).
Slave: DT Start of Vector	Const, MI, ML, DW	The start of the vector of bytes to be read (data source).
Slave: DT Offset in Vector	Const, MI, ML, DW	Offset from the Slave: DT Start of Vector
Master: DT Start of Vector	Const, MI, ML, DW	This is the start of a vector of bytes that will contain the data read from the slave.
Master: DT Offset in Vector	Const, MI, ML, DW	Offset from the Master: DT Start of Vector
Read: DT Vector Length	Constant or MI	The vector length. **Note ●** A MODBUS command cannot read/write more than 242 DT bytes at one time. In addition, 0 is not a legal length.
Status Messages	MI	Shows a message number. To check status and diagnose errors, check the MODBUS Status Messages.
Total Sessions	DW	This is the number of times the master PLC will attempt to access the slave device. Note that this is a simple incremental counter. Initialize it by storing 0 into the selected DW.
Acknowledgements	DW	This is the number of times the slave device answers.

Write to Data Table

Below, a MODBUS master writes to data tables in Slave ID 1. Bytes 140-159 are written from the master into bytes 24-43 in the slave's data tables.


Determines the Slave from which the data tables are read.
Determines the start byte of the source vector to be read in the slave's data tables.
Contains the value of the offset from the source vector within the data tables.
Determines, within the master's data tables, the start byte of the target vector used to store the slave data.
Contains the value of the offset from the target vector within the data tables.
Determines the length of the source vector--the number of bytes to be read from the slave's data tables.
Slave 1: Data Tables
Bytes 18 19 20 21 22 23 24 ... 43 44
Master: Data Tables
Bytes 117 118 119 120 ... 140 ... 159 160 161
EN ENO
MODBUS READ DT MODBUS_1
Slave ID 1
D# 20 Slave: DT Start
D# 4 Slave: DT Offset
D# 120 Master: DT Start
D# 20 Master: DT
D# 20 Master: DT
MI 0 Error Status
DW 0 Total Sessions
DW 1 Acknowledgeme
The vector to be read is 20 bytes long.
20 data table bytes, from 24 to 43, are read from Slave 1. The master stores the data into bytes 140-159.

Parameter	Type	Function
Slave ID	Constant or MI	The ID of the slave device to which the data will be written (data target).
Slave: DT Start of Vector	Const, MI, ML, DW	The start of the vector of bytes to be written into (data target).
Slave: DT Offset in Vector	Const, MI, ML, DW	Offset from the Slave: DT Start of Vector
Master: DT Start of Vector	Const, MI, ML, DW	This is the start of a vector of bytes, in the master, that will contain the data to be written to the slave (data source)
Master: DT Offset in Vector	Const, MI, ML, DW	Offset from the Master: DT Start of Vector
Read: DT Vector Length	Constant or MI	The vector length. **Note** ● A MODBUS command cannot read/write more than 242 DT bytes at one time. In addition, 0 is not a legal length.
Status Messages	MI	Shows a message number. To check status and diagnose errors, check the MODBUS Status Messages.
Total Sessions	DW	This is the number of times the master PLC will attempt to access the slave device. Note that this is a simple incremental counter. Initialize it by storing 0 into the selected DW.
Acknowledgements	DW	This is the number of times the slave device answers.

Read/Write Mixed Data via MODBUS

The Read/Write Mixed Data function enables you to combine Read & Write operations in a single MODBUS command, transferring data between master and slave controllers, without using standard MODBUS commands and addressing conventions.

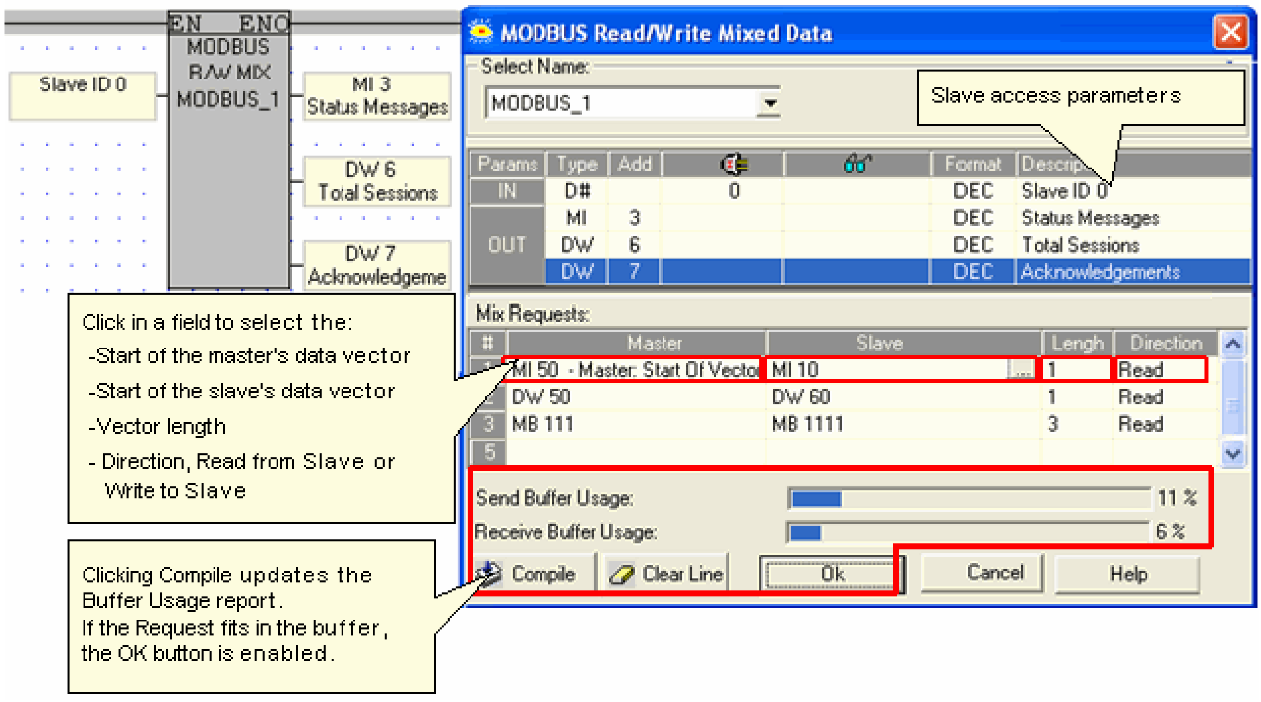
This is **not** a standard MODBUS function. Read/Write Mixed Data is compatible with Unitronics' Vision controllers **only** .

Note that this function is compatible with O/S 401 and higher, and is not supported for V120-12-xxx.

Each function can contain both Read and Write Mix Requests. Each request may be for a different data type. Your data request must include:

- Master and Slave operand addresses
- Length of vector
- Direction: Read or Write

After you add a request, the OK button is disabled. Click the Compile button to see current buffer status; if the buffer contains less than the maximum number of bytes, the OK is enabled.



Select Name: MODBUS_1

Slave access parameters

Params	Type	Add			Format	Description
IN	D#		0		DEC	Slave ID 0
OUT	MI	3			DEC	Status Messages
	DW	6			DEC	Total Sessions
	DW	7			DEC	Acknowledgements

Mix Requests:

#	Master	Slave	Length	Direction
1	MI 50 - Master: Start Of Vector	MI 10	1	Read
2	DW 50	DW 60	1	Read
3	MB 111	MB 1111	3	Read
5				

Send Buffer Usage: 11 %

Receive Buffer Usage: 6 %

Click in a field to select the:
-Start of the master's data vector
-Start of the slave's data vector
-Vector length
- Direction, Read from Slave or Write to Slave

Clicking Compile updates the Buffer Usage report.
If the Request fits in the buffer, the OK button is enabled.

Read Write Limitations

Only the following data types may be used in Mixed Data requests: MI, ML, DW, MB, I and O.

Registers: may only be read/written to the same data type.

Booleans: Inputs cannot be written to.

Booleans, Read Write		
I	→	MB, O
O	↔	MB, O
MB	↔	MB, O

Registers, Read Write		
MI	↔	MI
ML	↔	ML
DW	↔	DW

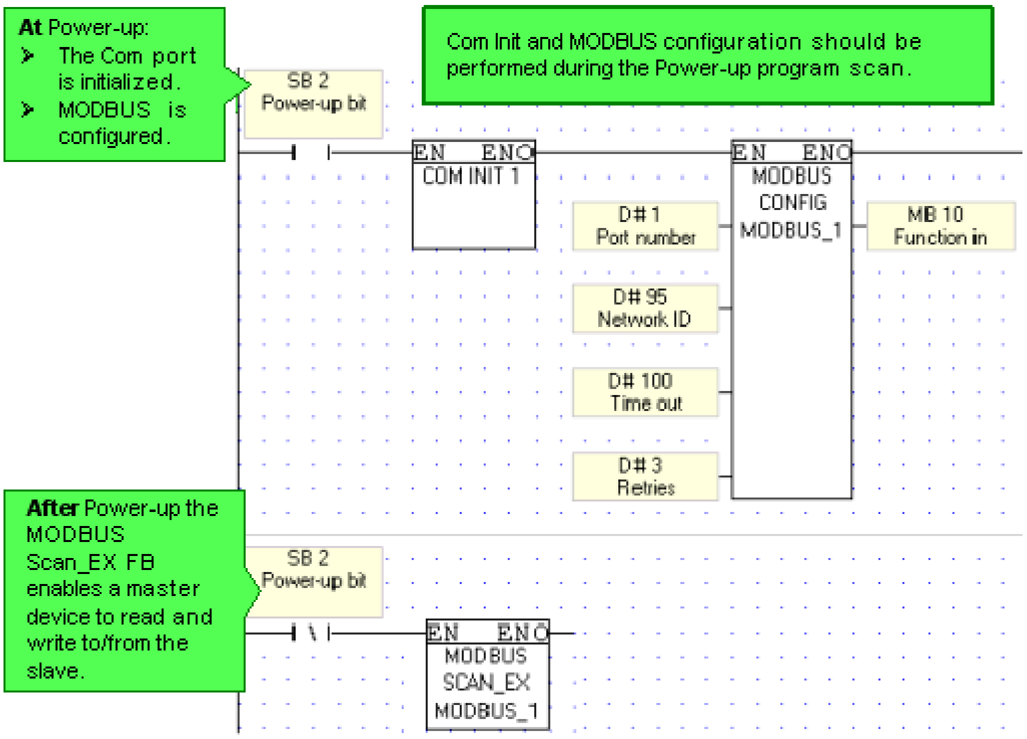
Send / Receive Buffers

The function uses two buffers, Send and Receive. Each buffer can contain a maximum of 500 bytes.

Configuring a MODBUS slave device

The Ladder section below shows what elements are necessary to enable a master device to read from a slave. Note that the MODBUS Scan_EX operation should **not** be performed during the initial program scan.

Note that you must use a condition (RLO) to activate the MODBUS Configuration and SCAN _EX.


At Power-up:
➢ The Com port is initialized.
➢ MODBUS is configured.
Com Init and MODBUS configuration should be performed during the Power-up program scan.
SB 2 Power-up bit
EN ENO COM INIT 1
EN ENO MODBUS CONFIG MODBUS_1
D# 1 Port number
D# 95 Network ID
D# 100 Time out
D# 3 Retries
MB 10 Function in
After Power-up the MODBUS Scan_EX FB enables a master device to read and write to/from the slave.
SB 2 Power-up bit
EN ENO MODBUS SCAN_EX MODBUS_1

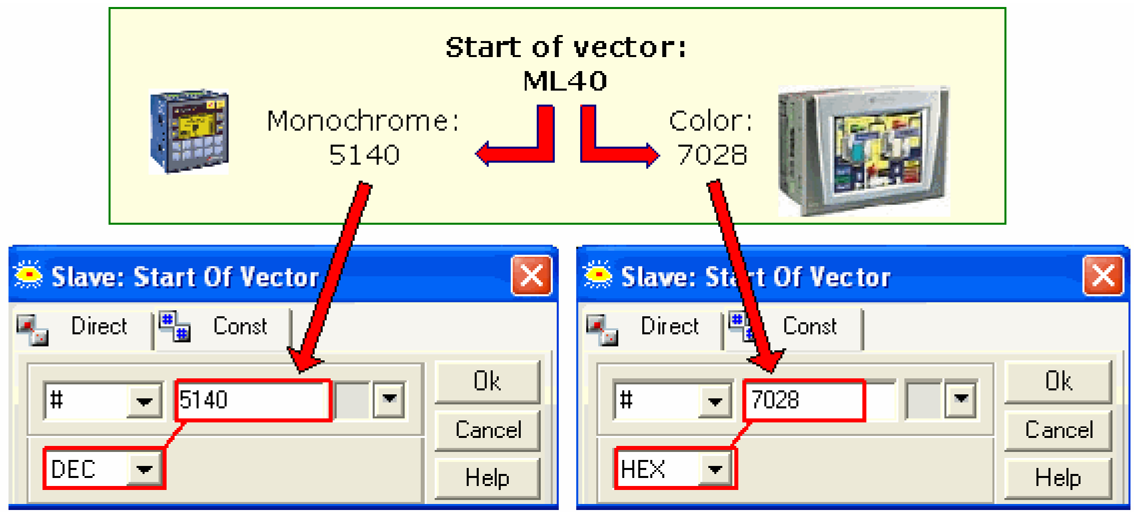
Slave Addressing

Monochrome / Color PLCs

The memory structure of monochrome screen PLCs is different from that of color screen PLCs. This is why each series has its own slave addressing scheme.

Note • The slave addresses given for the **monochrome** series are **decimal** values.

The slave addresses given for the **color** series are **hexadecimal** values.



Slave Addresses

Standard Vision Division

Coils		MODBUS Command Number	
Pointer Value From:	Operand type	Read	Write
0000	MB 0-2999	#01 Read Coils	#15 Force Coils
3000	SB		#15 Force Coils
4000	I (read-only)		Read-only
5000	O		#15 Force Coils
6000	T (read-only)		Read-only
7000	C (read-only)		Read-only
8000	MB 3000-4095		

Note • Note that in order to access MBs 3000-4095, you address as follows:

to access MB 3012, request slave address 8012.

Registers			MODBUS Command Number	
Pointer Value From:	Operand type	Register size	Read	Write
0000	MI	16 bit	# 03 Read Holding Registers	# 16 Preset Holding Registers
4000	SI	16 bit		
5100	ML	32 bit		
6100	SL	32 bit		
6300	MDW	32 bit		
6700	SDW	32 bit		
6900	Timer preset	32 bit		
7300	Timer current	32 bit		
7700	MF	32 bit		
7800	Counter Preset	16 bit		
7900	Counter Current	16 bit		

Enhanced Vision Series

Coils		MODBUS Command Number	
Pointer Value From (hex):	Operand type	Read	Write
0000h	MB 0	#01 Read Coils	#15 Force Coils
3000h	XB		#15 Force Coils
4000h	O		#15 Force Coils
5000h	SB		Read-only
6000h	I (read-only)		#15 Force Coils
7000h	T (read-only)		Read-only
8000h	C (read-only)		Read-only

Registers			**MODBUS Command Number**	
Pointer Value From (hex):	Operand type	Register size	Read	Write
0000h	MI	16 bit	# 03 Read Holding Registers	# 16 Preset Holding Registers
3000h	XI			
9000h	SI	16 bit		
5000h	XL			
6000h	XDW			
7000h	ML	32 bit		
A000h	SL	32 bit		
8000h	MDW	32 bit		
B000h	SDW	32 bit		
C000h	Timer preset	32 bit		
D000h	Timer current	32 bit		
4000h	MF	32 bit		
E000h	Counter Preset	16 bit		
F000h	Counter Current	16 bit		

Examples

The examples below show that:

- MODBUS addressing systems start at 1.
- Vision addressing starts at 0.

Bit Operands

Read a 10-bit vector of inputs in a slave Vision controller, starting at Input 20, via Read Coils (MODBUS COMMAND #1)

- Vision PLC as the MODBUS master to Monochrome PLC

  In VisiLogic's Read Coils FB, set the Slave: Start of Vector parameter to 4020 (DEC), and the Read: Vector Length parameter to 10. Within the slave Vision controller, VisiLogic will read I 20 - I 29.

- Vision PLC as the MODBUS master to Color PLC

  In VisiLogic's Read Coils FB, set the Slave: Start of Vector parameter to 6014h (HEX), and the Read: Vector Length parameter to 10. Within the slave Vision controller, VisiLogic will read I 20 - I 29.

- SCADA as the MODBUS master to Monochrome PLC
  In the SCADA application, set the Slave: Start of Vector parameter to 34021(30001 + 4000 + 20), and the Read: Vector Length to 10, enabling the Master device to read I 20 - I 29 within the slave Vision controller.

- **SCADA as the MODBUS master to Color PLC**
  Convert the HEX address to DEC
  In the SCADA application, set the Slave: Start of Vector parameter to 54597(24576 (6000h) + 20), and the Read: Vector Length to 10, enabling the Master device to read I 20 - I 29 within the slave Vision controller.

Write a 3-bit vector of outputs in a slave Vision controller, starting at Output 8, via Force Coils (MODBUS COMMAND #15)

- Vision PLC as the MODBUS master to Monochrome PLC
  In VisiLogic's Force Coils FB, set the Slave: Start of Vector parameter to 5008, and the Force: Vector Length parameter to 3. Within the slave Vision controller, the master will force the status of O 8 - O 10.

- **Vision PLC as the MODBUS master to Color PLC**
  In VisiLogic's Force Coils FB, set the Slave: Start of Vector parameter to 4008h (HEX), and the Force: Vector Length parameter to 3. Within the slave Vision controller, the master will force the status of O 8 - O 10.

- SCADA as the MODBUS master to Monochrome PLC
  In the SCADA application, set the Slave: Start of Vector parameter to 35009 (30001 + 5000 + 8) and the Force: Vector Length parameter to 3, enabling the Master device to write to O 8 - O 10 within the slave Vision controller.

- **SCADA as the MODBUS master to Color PLC**
  Convert the HEX address to DEC.
  In the SCADA application, set the Slave: Start of Vector parameter to 46393 (30001 + 16384(4000h) + 8) and the Force: Vector Length parameter to 3, enabling the Master device to write to O 8 - O 10 within the slave Vision controller.

Registers

Read a 9-register long vector of **16-bit integers** in a slave Vision controller, starting at MI 32, via Read Holding Registers (MODBUS COMMAND #03)

- Vision PLC as the MODBUS master
  In VisiLogic's Read Holding Registers FB, set the Slave: Start of Vector parameter to 32, and the Read: Vector Length parameter to 9. Within the slave Vision controller, VisiLogic will read MI 32 - MI 41.
- SCADA as the MODBUS master
  In the SCADA application, set the Slave: Start of Vector parameter to 40033 (40001 + 0000 + 3), and the Read: Vector Length parameter to 9, enabling the Master device to read MI 32 - MI 41 within the slave Vision controller.

Note • If, within the Slave: Start of Vector parameter, the selected register type is a **32-bit double register** (slave addresses 5100 and greater) the preset vector length must be **doubled** as well.
If, for example in the VisiLogic Preset Holding Registers FB:

- Slave: Start of Vector parameter is set to 6300, and
- You wish to preset 4 registers, for a total of 16 bytes
- You must set the Preset Vector length to 8.

Note that this means that, in these cases, the **Preset: Vector Length** parameter will always be an even number.

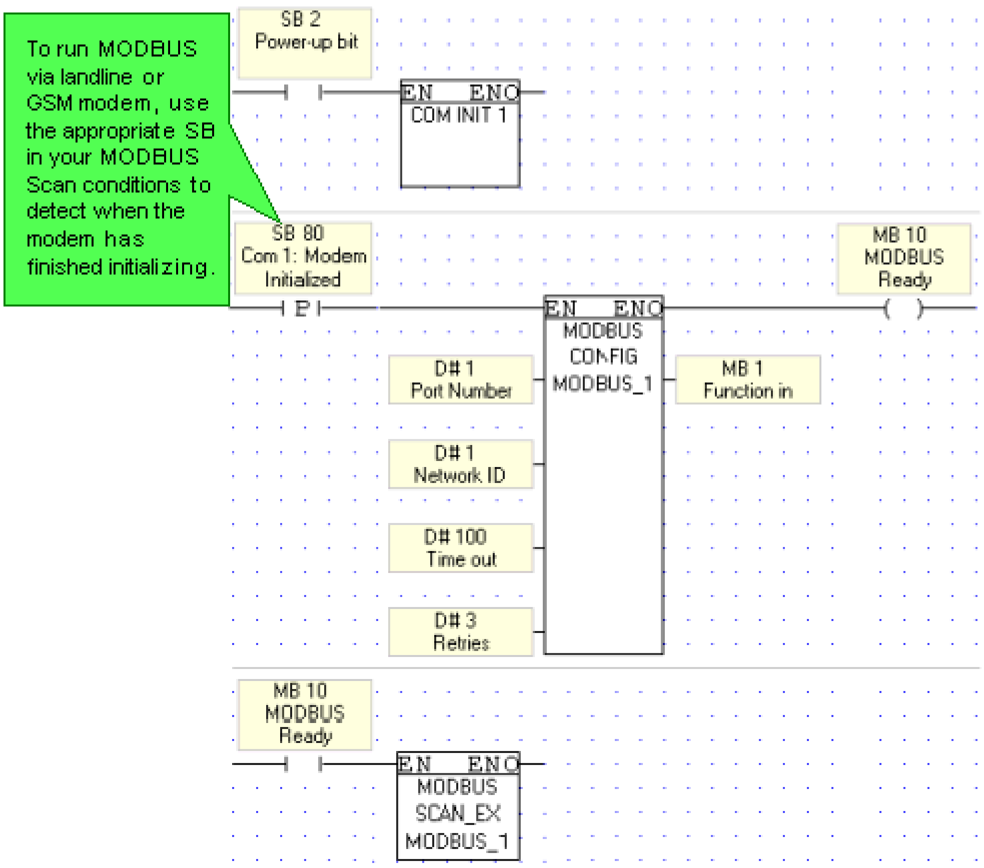
Read a 9-register long vector of **32 -bit integers** in a slave Vision controller, starting at SL 32, via Preset Holding Registers (MODBUS COMMAND #16)

- Vision PLC as the MODBUS master
  In VisiLogic's Preset Holding Registers FB, set the Slave: Preset Vector parameter to 6132, and the Read: Vector Length parameter to 18 (2x9, in order to fit the 32-bit SL registers). Within the slave Vision controller, VisiLogic will read SL 32 - SL 41.
- SCADA as the MODBUS master
  In the SCADA application, set the Slave: Start of Vector parameter to 406133, and the Read: Vector Length parameter to 18, enabling the Master device to read SL 32 - SL 41 within the slave Vision controller.

Write a 6-register long vector of **16-bit integers** in a slave Vision controller, starting at MI 32, via Preset Holding Registers (MODBUS COMMAND #16)

- Vision PLC as the MODBUS master
  In VisiLogic's Preset Holding Registers FB, set the Slave: Start of Vector parameter to 32, and the Preset: Vector Length parameter to 6. Within the slave Vision controller, VisiLogic will write to MI 32 - MI 37.
- SCADA as the MODBUS master
  In the SCADA application, set the Slave: Start of Vector parameter to 40033, and the Read: Vector Length parameter to 6, enabling the Master device to write to MI 32 - MI 37 within the slave Vision controller.

MODBUS via GSM or Standard Modem


To run MODBUS via landline or GSM modem, use the appropriate SB in your MODBUS Scan conditions to detect when the modem has finished initializing.
SB 2 Power-up bit
EN ENO COM INIT 1
SB 80 Com 1: Modem Initialized
P
EN ENO MODBUS CONFIG MODBUS_1
MB 10 MODBUS Ready
D# 1 Port Number
MB 1 Function in
D# 1 Network ID
D# 100 Time out
D# 3 Retries
MB 10 MODBUS Ready
EN ENO MODBUS SCAN_EX MODBUS_1

Note • When MODBUS (Serial) is configured to a port linked to a modem, the MODBUS function checks SI 100 Maximum Time Delay between characters (units 2.5ms) MODBUS + Modem. If SI 100 = 1, a time interval of up to 2.5 msecs is permitted between characters, if SI 100 contains 2, the permitted interval is 5 msecs (n x 2.5 = interval. Note that the application must update SI 100 before the MODBUS configuration is activated.

MODBUS Status Operands

All of the Status operands linked to MODBUS FBs should be assigned Power-up Values; bits should be reset, and registers initialized to 0.

MODBUS: Configuration FB Status Operand
All MODBUS operations run through a MODBUS configuration placed in the master device's program.

Function in Progress Shows status of master's **MODBUS Configuration**	**MB**	**Turns ON when:** • A master Vision initiates MODBUS communication. • Remains ON during the MODBUS session.	**Turns OFF when** • The **MODBUS: Configuration** is activated. • An answer is received from a slave. • The TimeOut defined in the **Configuration** is exceeded. • Certain Status Messages are given

MODBUS Operation Status Operands
When you place MODBUS operations in your application (Force, Read, Preset, and Loopback commands), you link the operands below. These show the status of MODBUS sessions.

Status Messages Shows status of master's data requests and the replies the master receives from the slaves	**MI**	• Automatically initialized to 0 when MODBUS operation is activated. • Updated at the end of each attempt to communicate via MODBUS. • Indicates status of **MODBUS** communications, according to the table below. Note that the current value always shows the most **recent** status.

#	**Status Message**
0	**Status OK**
1	**Unknown Command Number** This is received from the slave device**.**
2	**Illegal Data Address** • Master: an invalid address is found by the master before a data request is sent to a slave. This may result, for example, when an MI is used to provide vector length. • Slave: The slave notifies the master that the data request command includes invalid addresses. • Slave--ScanEX: When ScanEX receives an input parameter in the 32-bit range (for example, 5100{ML}), it automatically takes double-register values. If, for example, ScanEX receives a Read Register(6) request for **5100**, it returns the values in 5100 **and** 5101. If, however, ScanEX receives Read Register(6) request for **5101**, it returns Error #2-- since 5101 provides the 'high' bytes of the 32-bit register, it is not a legal address.
3	**Slave to Master: Illegal Data Type Quantity** Number of operands requested by user exceeds the maximum **Note •** A MODBUS command cannot read more than 124 16-bit integers, 62 double registers, 62 float registers, or 1900 bit operands at one time. In addition, 0 is not a legal vector length.

4	**Master--Time Out** This occurs if the master has been waiting for a slave response for an amount of time exceeding the Time-out set in the Configuration. If this error occurs, check **Time Out, Baud Rate, and Distance**
5	**No Communication** The MODBUS session cannot be established.

Note • Messages 4 & 5. **TimeOut** and **Number of Retries** are defined in the **Configuration**. A Retry is an attempt to establish a MODBUS session.
If, for example, TimeOut is defined as 2 seconds, and number of Retries as 3:
- the controller will try to establish the session once, and will continue to try for 2 seconds.
- If the first attempt fails, the **Status Message value will be 4**, Master TimeOut.
-The controller will try twice more, for a total of 3 retries over 6 seconds.
- If all attempts fail, the **Status Message value will be 5**.
-If any attempt succeeds, the Status Message will be 0.

*** 6**	**Master-slave data incorrectly synchronized**
*** 7**	**Master-slave data incorrectly synchronized**
8	**Master to application: Illegal Data Type Quantity** Number of operands requested by user exceeds the maximum permitted for that FB operation in the master. **Note •** A MODBUS command cannot read more than 124 16-bit integers, 62 double registers, 62 float registers, or 1900 bit operands at one time. In addition, 0 is not a legal vector length.
9	**Slave ID =0** An attempt does to communicate with Slave ID 0.
10	**Incorrect CRC**
*** 11**	**Master-slave data incorrectly synchronized**

***** Messages 6, 7, and 11mean that the master has found incompatible elements in the data sent between master and slave.

Total Sessions	DW	This is the total number of times the master PLC attempts to access the slave device, whether the attempt is successful or not, including Retries. Note that this is a simple incremental counter. • This must be initialized by the user, by storing 0 into the selected DW.
Acknowledgements	DW	This is the number of times the slave device answers. • This must be initialized by the user, by storing 0 into the selected DW.

Time Out, Baud Rate, and Distance

The default Time Out set in the Configuration is one second. In certain cases, the length of the network may cause this to be exceeded. You can calculate a Time Out value by using the following formula, and use it in the Configuration.

$$\text{Time Out} = \frac{(\text{TX bytes} + \text{RX bytes} + 7) \times 1000}{\text{baud rate} / (\text{data bits} + \text{start bits} + \text{stop bits}) * 10\text{ms}}$$

SI	Description	Value
100	Maximum Time Delay between characters (units 2.5ms) MODBUS + Modem	When MODBUS (Serial) is configured to a port linked to a modem, the MODBUS function checks SI 100. If SI 100 = 1, a time interval of up to 2.5 msecs is permitted between characters, if SI 100 contains 2, the permitted interval is 5 msecs (n x 2.5 =interval. Note that: - The power-up value is 1, - the application must update SI 100 before the MODBUS configuration is activated.

MODBUS (IP)

MODBUS IP Overview

Converting Projects: Vision Divisions

The memory structure of Standard Vision controllers is different from that of Enhanced. Note that if you convert projects, you must make changes according to the information given in the Slave Address tables.

If your controller comprises an Ethernet card, you can use MODBUS IP commands with any connected device that supports the MODBUS protocol.

Within a MODBUS network, you can use standard MODBUS commands to read and write bit and register data; you can also read and write data to Vision controller Data Tables.

Any controller in the network may function as either master or slave.

Unitronics currently supports both TCP and UDP, as explained in the topic About Ethernet. This topic also contains general information about Ethernet, IP addressing, sockets, and ports.

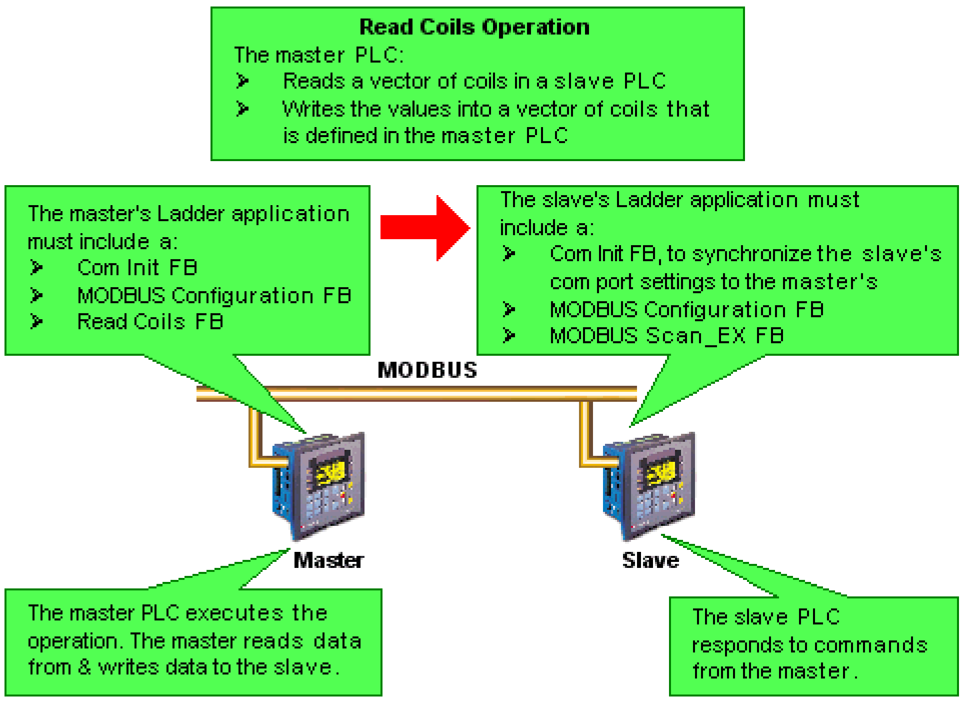
Specific information on implementing Ethernet is provided in the topic Using Ethernet.

Using MODBUS

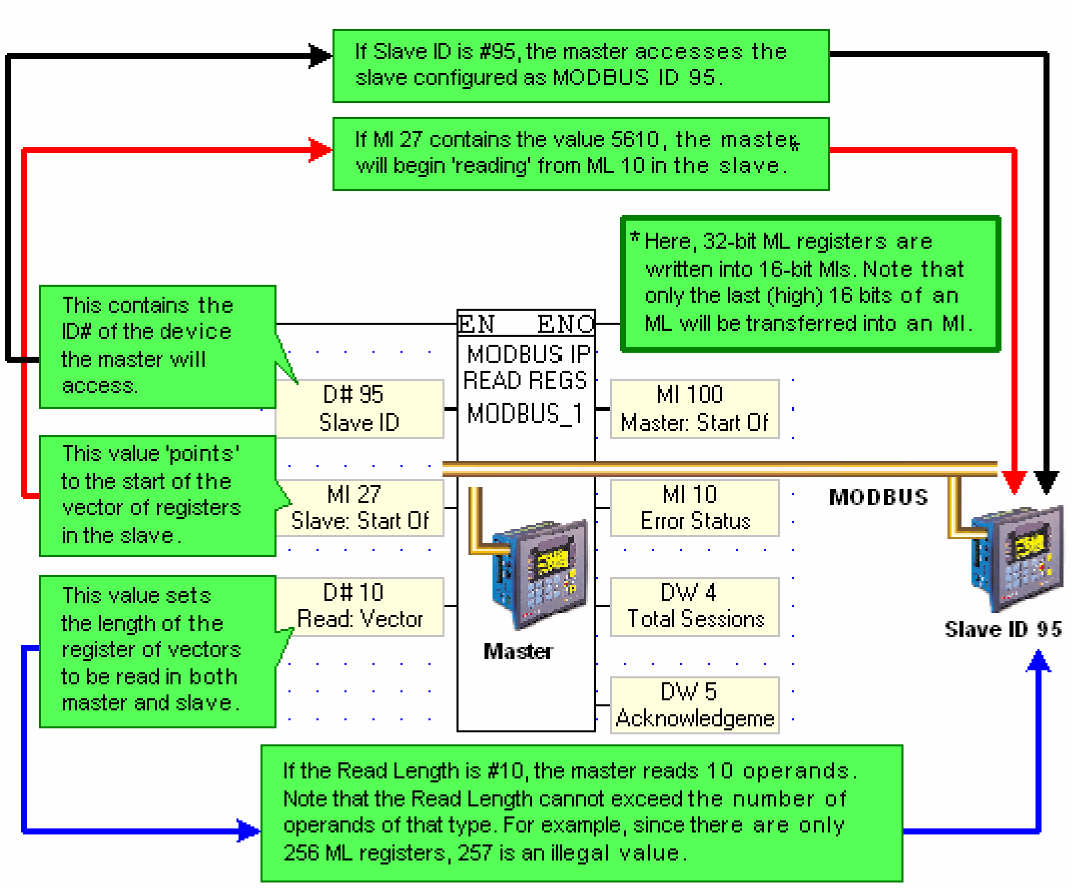
Before using a MODBUS IP operation in your application, you must:

- Assign IP addresses to both master and slave devices. This is done by placing Ethernet Card Init FBs in the ladder application of both master and slave.
- Include at least 1 MODBUS Configuration FB in the ladder application of both master and slave.
- The condition that activates the Configuration must turn ON for a single program cycle (positive transition recommended)
However, the MODBUS configuration must be scanned during every program cycle--after the Configuration is activated. One way to ensure this is by placing the configuration in the first subroutine of the main module.
- Enable slave devices to be accessed by placing a Scan FB in the slave's Ladder application.

The figure below shows the elements required to carry out a Read Coils Operation.


Read Coils Operation
The master PLC:
➢ Reads a vector of coils in a slave PLC
➢ Writes the values into a vector of coils that is defined in the master PLC
The master's Ladder application must include a:
➢ Com Init FB
➢ MODBUS Configuration FB
➢ Read Coils FB
The slave's Ladder application must include a:
➢ Com Init FB, to synchronize the slave's com port settings to the master's
➢ MODBUS Configuration FB
➢ MODBUS Scan_EX FB
MODBUS
Master
Slave
The master PLC executes the operation. The master reads data from & writes data to the slave.
The slave PLC responds to commands from the master.

Note that the operand addresses in slave PLCs are indirect addresses (pointers).


If Slave ID is #95, the master accesses the slave configured as MODBUS ID 95.
If MI 27 contains the value 5610, the master* will begin 'reading' from ML 10 in the slave.
* Here, 32-bit ML registers are written into 16-bit MIs. Note that only the last (high) 16 bits of an ML will be transferred into an MI.
This contains the ID# of the device the master will access.
This value 'points' to the start of the vector of registers in the slave.
This value sets the length of the register of vectors to be read in both master and slave.
EN ENO
MODBUS IP READ REGS MODBUS_1
D# 95 Slave ID
MI 27 Slave: Start Of
D# 10 Read: Vector
MI 100 Master: Start Of
MI 10 Error Status
DW 4 Total Sessions
DW 5 Acknowledgeme
MODBUS
Master
Slave ID 95
If the Read Length is #10, the master reads 10 operands. Note that the Read Length cannot exceed the number of operands of that type. For example, since there are only 256 ML registers, 257 is an illegal value.

Note that a MODBUS master can broadcast to the MODBUS network by writing to Slave ID # 0. To do this, indirectly address the Slave ID to a register, and write 0 to that register.

FB Operations

MODBUS IP Operations are located on the FBs menu.

MODBUS: Configuration

MODBUS: Scan

MODBUS: Read Coils (1)

MODBUS: Read Inputs (2)

Read Holding Registers (3)

Read Float Registers (3)

Read Input Registers (4)

Read Float Input Registers (4)

Force Coil (5)

Preset Holding Register (6)

Force Coils (15)

Preset Holding Registers (16)

Preset Float Registers (16)

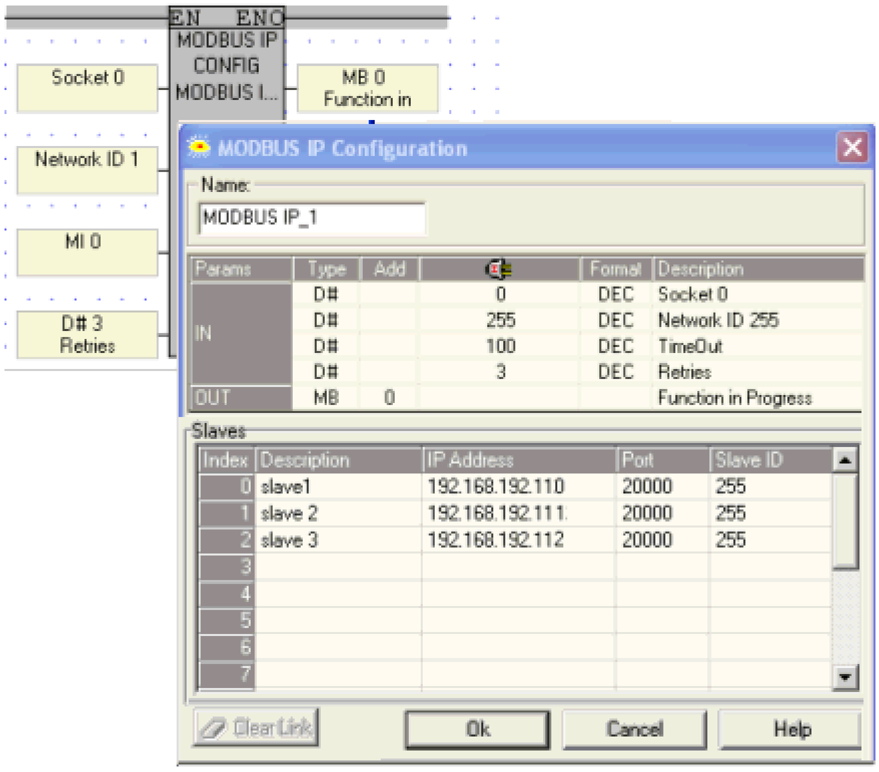
Read/Write Mixed Data (non-standard MODBUS command)

Read/Write to Data Tables

MODBUS Status Operands

MODBUS: Configuration

A MODBUS Configuration FB must be included in both master and slave Ladder applications as shown below. MODBUS IP Operations are located on the FBs menu.



Params	Type	Add		Format	Description
IN	D#		0	DEC	Socket 0
	D#		255	DEC	Network ID 255
	D#		100	DEC	TimeOut
	D#		3	DEC	Retries
OUT	MB	0			Function in Progress

Index	Description	IP Address	Port	Slave ID
0	slave1	192.168.192.110	20000	255
1	slave 2	192.168.192.111	20000	255
2	slave 3	192.168.192.112	20000	255
3				
4				
5				
6				
7				

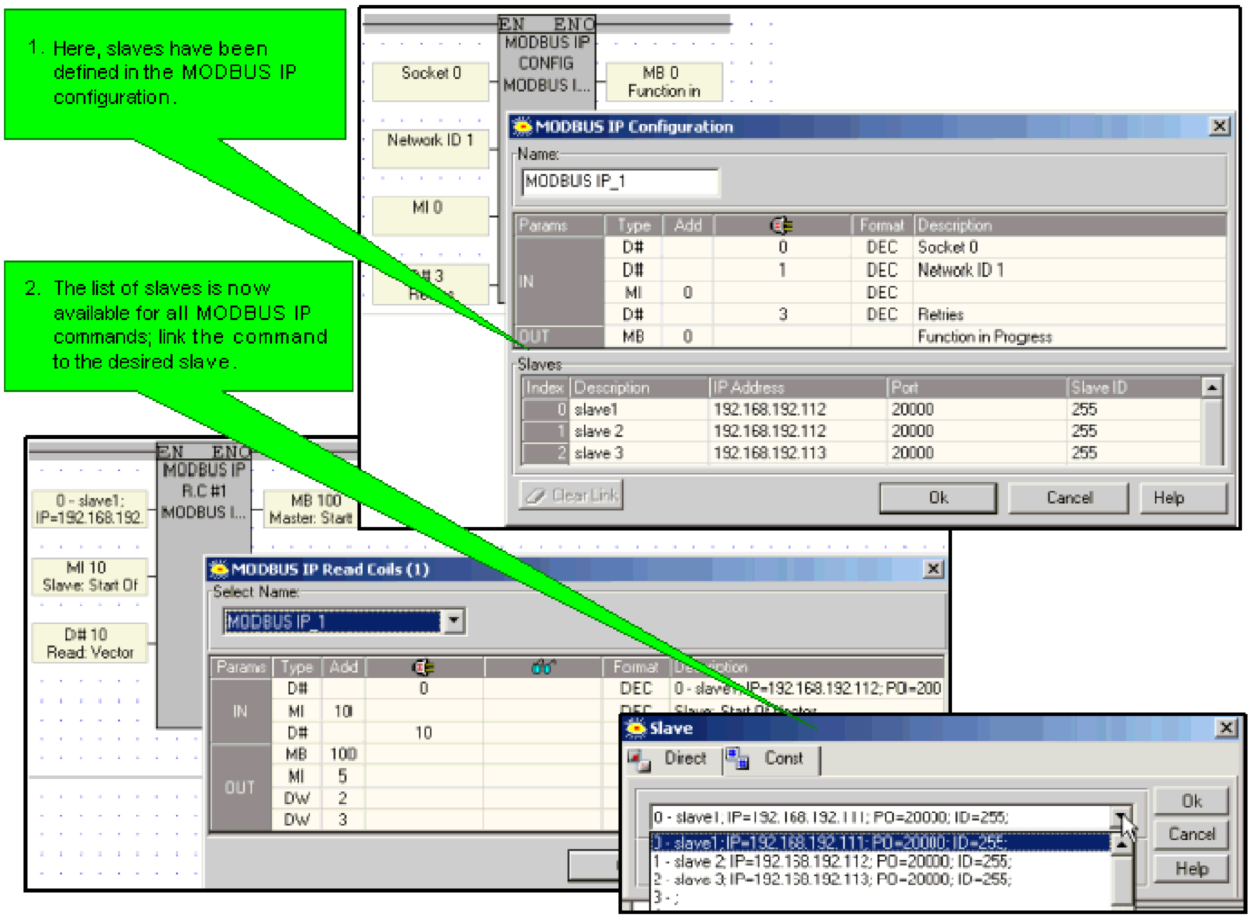
Parameter	Type	Function
Port Number	Constant	Click the drop-down arrows to view available ports; click the port you want to use.
Network ID	Constant	This number identifies the device on the network. You can either assign an ID via an MI, or directly via a constant number. The unit ID range is from 0-255. Do not assign the same ID number to more than one device.
Time out	Constant or MI	This is the amount of time a master device will wait for an answer from a slave. Time out units are defined in 10 msecs; a Time out value of 100 is equal to 1 second.
Retries	Constant or MI	This is the number of times a device will try to send a message.
Function in Progress	MB	This bit is ON when MODBUS is active. Use this as a condition bit for MODBUS operations to avoid communication conflicts.

Note • Indirectly addressed parameters in a MODBUS Configuration FB are only read when the Configuration is called. Since a Configuration is generally called as a power-up task, if, for example Retries has been indirectly addressed, and the linked MI is updated, the new value will **not** be read into the Configuration. The value will only be updated

	until the Configuration is called.
•	While a master attempts to send a command, the Function In Progress bit is ON. The number of attempts that the master will make is the number in Retries +1, where '1' is the initial access attempt.
•	When a master attempts to access a slave device, and the slave does not answer,- the Function In Progress bit will turn ON. This bit will remain on according to the following: (the number of retries + 1) x (Time Out), where '1' is the initial access attempt. Note that the Time Out parameter is in units of 10 msec.

Vision Slaves

In order to access Vision Controllers as slave devices and implement MODBUS commands, you must enter the IP addresses of the slave devices in the MODBUS IP configuration. This means that you must first assign IP addresses to each slave. This is done via the Ethernet Card Init FB, which must be configured as described in the topic Using_Ethernet.



1. Here, slaves have been defined in the MODBUS IP configuration.
2. The list of slaves is now available for all MODBUS IP commands; link the command to the desired slave.

MODBUS IP Configuration

Name: MODBUS IP_1

Params	Type	Add		Format	Description
IN	D#		0	DEC	Socket 0
	D#		1	DEC	Network ID 1
	MI	0		DEC	
	D#		3	DEC	Retries
OUT	MB	0			Function in Progress

Slaves

Index	Description	IP Address	Port	Slave ID
0	slave1	192.168.192.112	20000	255
1	slave 2	192.168.192.112	20000	255
2	slave 3	192.168.192.113	20000	255

MODBUS IP Read Coils (1)

Select Name: MODBUS IP_1

Params	Type	Add			Format	Description
IN	D#		0		DEC	0 - slave1; IP=192.168.192.112; PO=200
	MI	10			DEC	Slave: Start Of Vector
	D#		10			
OUT	MB	100				
	MI	5				
	DW	2				
	DW	3				

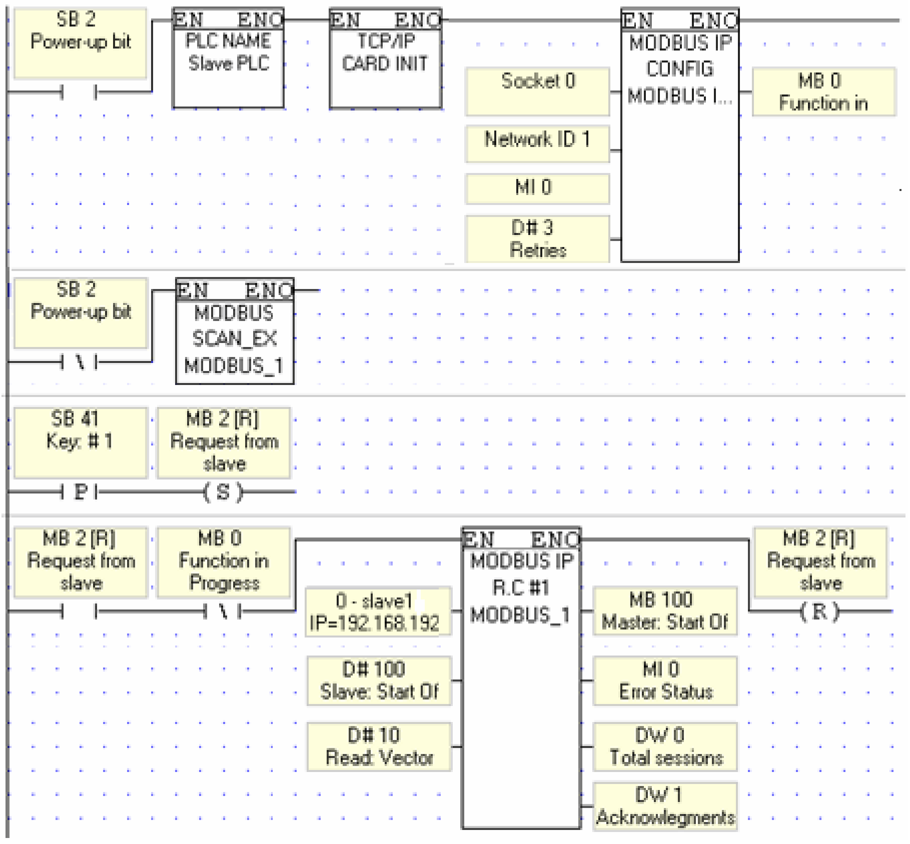
Slave

0 - slave1; IP=192.168.192.111; PO=20000; ID=255;
0 - slave1; IP=192.168.192.111; PO=20000; ID=255;
1 - slave 2; IP=192.168.192.112; PO=20000; ID=255;
2 - slave 3; IP=192.168.192.113; PO=20000; ID=255;
3 - ;

Note •	Slave IP addresses can also be linked to an MI vector, note that the vector is 4 MIs long. The low byte of each MI provides the number for an octet within the IP address. If, for example, the IP address is linked to MI 0, and the low bytes of MI 0 to MI 3 contain the values 192, 198, 192, 45, the IP address

will be 192.198.192. 45.

The Ladder application below enables the controller act as a MODBUS master and read coils in a slave PLC. The same PLC can also act as a slave, if a Scan_EX operation is included in the application.

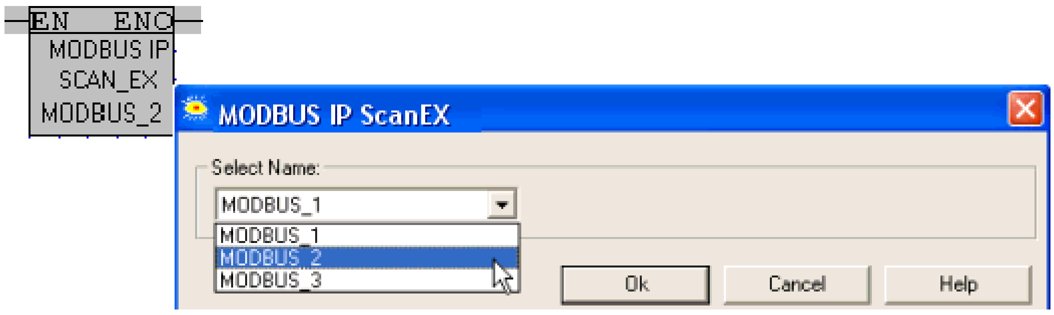


Status Operands

When you place MODBUS operations in your application (Force, Read, Preset, and Loopback commands), you link operands that show the status of MODBUS sessions. Use these to troubleshoot problems.

MODBUS: ScanEX and Scan

Scan_EX enables a master device to access a slave PLC. A Scan_EX must be included in the slave application. MODBUS IP Operations are located on the FBs menu.



Note • Scan_Ex is recommended for new applications.

About Scan and Scan_EX

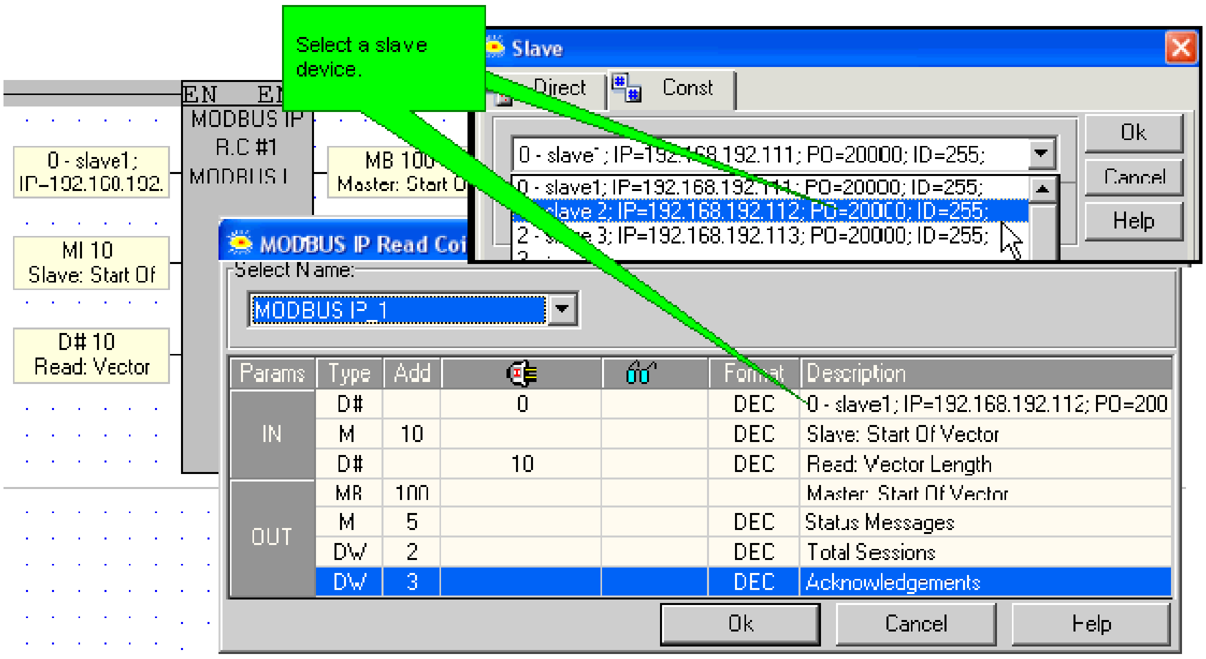
MODBUS Versions **previous to V2.01** offered only the **Scan** FB. Scan is still supported for older, working applications. When MODBUS operations accessed double registers (5100 addresses and higher), using odd addresses, such as 5101, there were incompatibility issues.

Scan_EX is recommended for new applications.

When ScanEX receives an input parameter in the 32-bit range (for example, 5100{ML}), it automatically takes double-register values.
If, for example, ScanEX receives a Read Register(6) request for **510<u>0</u>**, it returns the values in 5100 **and** 5101. If, however, ScanEX receives Read Register(6) request for **510<u>1</u>**, it returns Status Message #2-- since 5101 provides the 'high' bytes of the 32-bit register, it is not a legal address.

Read Coils (1)

Use this command to read the status of a selected group of coils and write them into a vector. The coil's status is written into a vector of MBs in the master PLC. MODBUS IP Operations are located on the FBs menu.



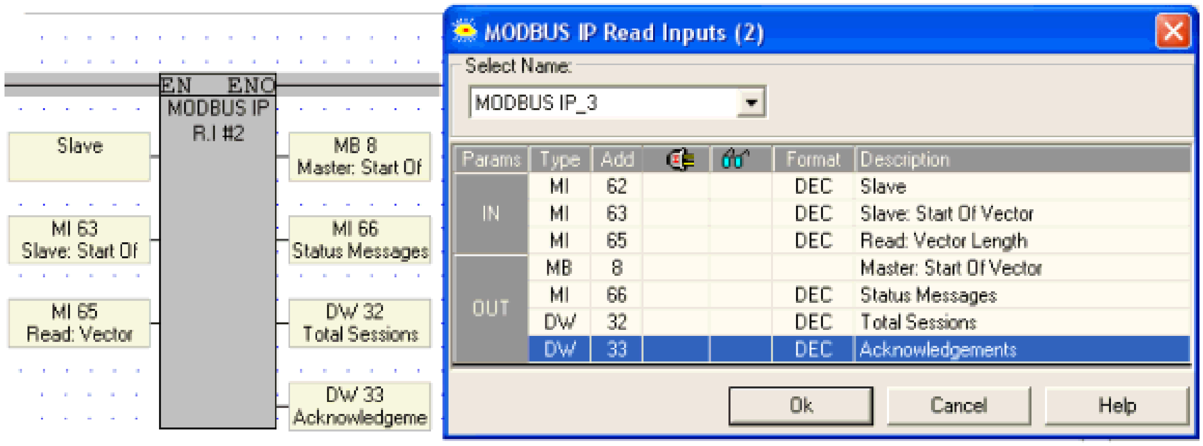
0 - slave˜; IP=192.168.192.111; PO=20000; ID=255;
0 - slave1; IP=192.168.192.111; PO=20000; ID=255;
1 - slave 2; IP=192.168.192.112; PO=20000; ID=255;
2 - slave 3; IP=192.168.192.113; PO=20000; ID=255;

Params	Type	Add			Format	Description
IN	D#		0		DEC	0 - slave1; IP=192.168.192.112; PO=200
	M	10			DEC	Slave: Start Of Vector
	D#		10		DEC	Read: Vector Length
OUT	MB	100				Master: Start Of Vector
	M	5			DEC	Status Messages
	DW	2			DEC	Total Sessions
	DW	3			DEC	Acknowledgements

Parameter	Type	Function
Slave ID	Constant or MI	The IP address of the slave device containing the coils to be read (data source).
Slave: Start of Vector	Const, MI, ML, DW	The start of the vector of coils to be read (data source). **Note** • Check topic Slave Address Tables.
Read: Vector Length	Constant or MI	The vector length. **Note** • A MODBUS command cannot read/write more than 1900 bit operands at one time. In addition, 0 is not a legal length.
Master: Start of Vector	MB	Shows a message number. To check status and diagnose errors, check the MODBUS Status Operands.
Status Messages	MI	Shows a message number. To check status and diagnose errors, check the MODBUS Status Operands.
Total Sessions	DW	This is the number of times the master PLC will attempt to access the slave device. Note that this is a simple incremental counter. Initialize it by storing 0 into the selected DW.
Acknowledgements	DW	This is the number of times the slave device answers.

Read Inputs (2)

Use this command to read the status of a selected group of inputs in a slave device and write them into a vector. The inputs's status is written into a vector of MBs in the master PLC. MODBUS IP Operations are located on the FBs menu.



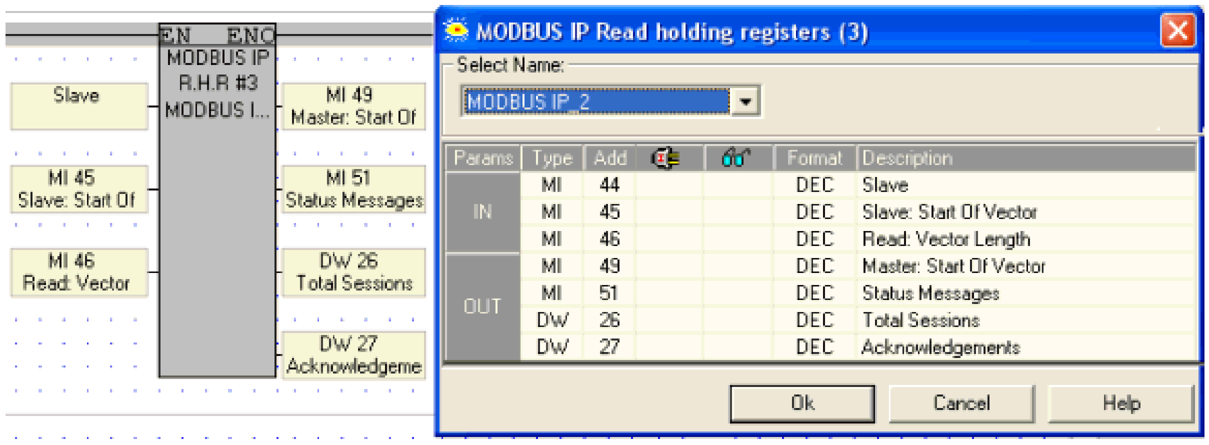
Select Name: MODBUS IP_3

Params	Type	Add			Format	Description
IN	MI	62			DEC	Slave
	MI	63			DEC	Slave: Start Of Vector
	MI	65			DEC	Read: Vector Length
OUT	MB	8				Master: Start Of Vector
	MI	66			DEC	Status Messages
	DW	32			DEC	Total Sessions
	DW	33			DEC	Acknowledgements

Parameter	Type	Function
Slave ID	Constant or MI	The IP address of the slave device containing the inputs to be read (data source).
Slave: Start of Vector	Const, MI, ML, DW	The start of the vector of inputs to be read (data source). **Note** • Check topic Slave Address Tables.
Read: Vector Length	Constant or MI	The vector length. **Note** • A MODBUS command cannot read/write more than 1900 bit operands at one time. In addition, 0 is not a legal length.
Master: Start of Vector	MB	This is the start of a vector of MBs that will contain the inputs' status in the master (data destination).
Status Messages	MI	Shows a message number. To check status and diagnose errors, check the MODBUS Status Operands.
Total Sessions	DW	This is the number of times the master PLC will attempt to access the slave device. Note that this is a simple incremental counter. Initialize it by storing 0 into the selected DW.
Acknowledgements	DW	This is the number of times the slave device answers.

Read Holding Registers (3)

Use this command to read the values of a selected group of registers in a slave PLC and write them into a defined vector of registers in the master. MODBUS IP Operations are located on the FBs menu.



MODBUS IP Read holding registers (3)

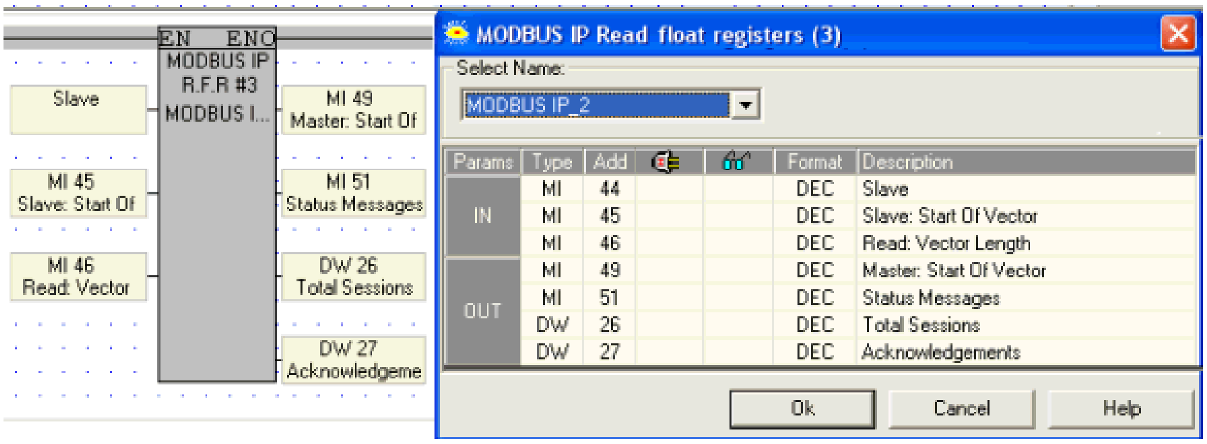
Select Name: MODBUS IP 2

Params	Type	Add			Format	Description
IN	MI	44			DEC	Slave
	MI	45			DEC	Slave: Start Of Vector
	MI	46			DEC	Read: Vector Length
OUT	MI	49			DEC	Master: Start Of Vector
	MI	51			DEC	Status Messages
	DW	26			DEC	Total Sessions
	DW	27			DEC	Acknowledgements

Parameter	Type	Function
Slave ID	Constant or MI	The IP address of the device containing the registers to be read (data source).
Slave: Start of Vector	Const, MI, ML, DW	The start of the vector of registers to be read (data source). **Note** • Check topic Slave Address Tables.
Read: Vector Length	Const, MI, ML, DW	The vector length **Note** • A MODBUS command cannot read more than 124 16-bit integers, 62 double registers, or 64 float registers at one time. In addition, 0 is not a legal length. • If, within the Slave: Start of Vector parameter, the selected register type is a **32-bit double register** (slave addresses 5100 and greater)the preset vector length must be doubled as well. If, for example: - Slave: Start of Vector parameter is set to 6300, and - You wish to preset 4 registers, for a total of 16 bytes - You must set the Preset Vector length to 8. Note that this means that, in these cases, the Preset: Vector Length parameter will always be an even number.
Master: Start of Vector	MI	This is the start of a vector of MIs that will contain the registers' values in the master (data destination).
Status Messages	MI	Shows a message number. To check status and diagnose errors, check the MODBUS Status Operands.
Total Sessions	DW	This is the number of times the master PLC will attempt to access the slave device. Note that this is a simple incremental counter. Initialize it by storing 0 into the selected DW.
Acknowledgements	DW	This is the number of times the slave device answers.

Read Flat Registers (3)

Use this command to read the values of a selected group of floating point registers in a slave device and write them into a defined vector of registers in the master. Values after the decimal point are rounded to the nearest whole value. MODBUS IP Operations are located on the FBs menu.



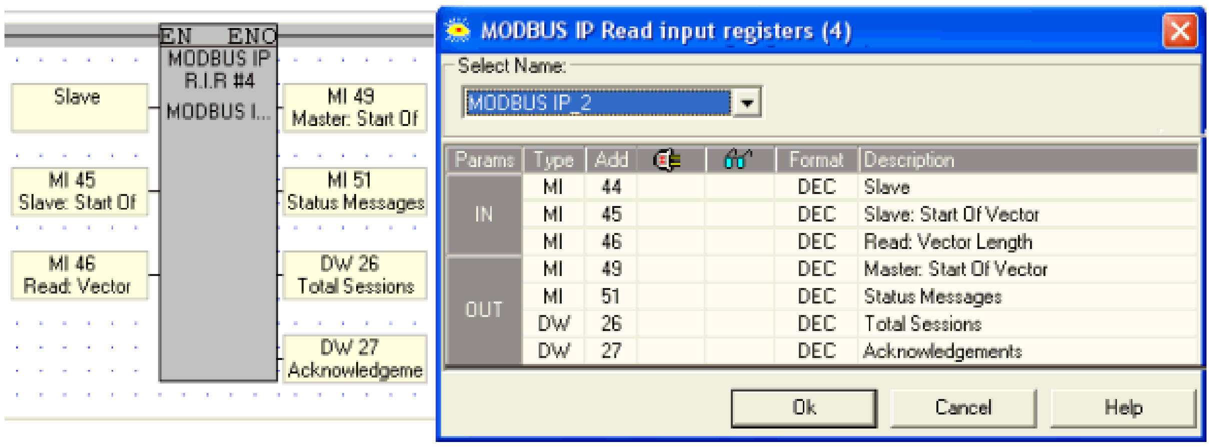
Select Name: MODBUS IP_2

Params	Type	Add			Format	Description
IN	MI	44			DEC	Slave
	MI	45			DEC	Slave: Start Of Vector
	MI	46			DEC	Read: Vector Length
OUT	MI	49			DEC	Master: Start Of Vector
	MI	51			DEC	Status Messages
	DW	26			DEC	Total Sessions
	DW	27			DEC	Acknowledgements

Parameter	**Type**	**Function**
Slave ID	Constant or MI	The IP address of the device containing the registers to be read (data source).
Slave: Start of Vector	Const, MI, ML, DW	The start of the vector of registers to be read (data source). **Note** • Check topic Slave Address Tables.
Read: Vector Length	Const, MI, ML, DW	The vector length **Note** • A MODBUS command cannot read more than 124 16-bit integers, 62 double registers, or 64 float registers at one time. In addition, 0 is not a legal length. • If, within the Slave: Start of Vector parameter, the selected register type is a **32-bit double register** (slave addresses 5100 and greater)the preset vector length must be doubled as well. If, for example: - Slave: Start of Vector parameter is set to 6300, and - You wish to preset 4 registers, for a total of 16 bytes - You must set the Preset Vector length to 8. Note that this means that, in these cases, the Preset: Vector Length parameter will always be an even number. • You can transpose 16 bits of a 32-bit double register value by turning SB 102 MODBUS Read Long ON in your program. SB 102 is OFF by default, and must be reset by the user program.
Master: Start of Vector	MI	This is the start of a vector of MIs that will contain the registers' values in the master (data destination).
Status Messages	MI	Shows a message number. To check status and diagnose errors, check the MODBUS Status Operands.
Total Sessions	DW	This is the number of times the master PLC will attempt to access the slave device. Note that this is a simple incremental counter. Initialize it by storing 0 into the selected DW.
Acknowledgements	DW	This is the number of times the slave device answers.

Read Input Registers (4)

Use this command to read the values of a selected group of registers in a slave PLC and write them into a defined vector of registers in the master. MODBUS IP Operations are located on the FBs menu.



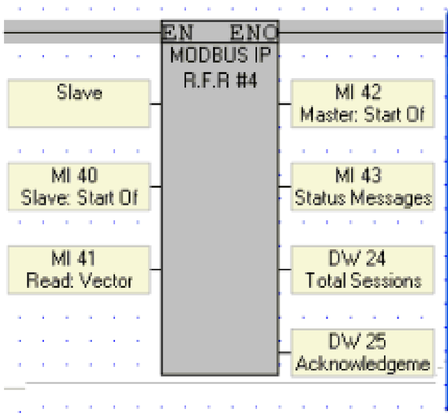
Select Name: MODBUS IP_2

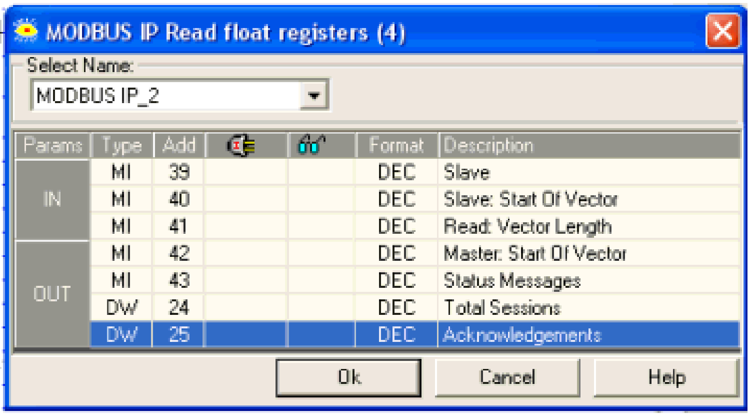
Params	Type	Add			Format	Description
IN	MI	44			DEC	Slave
	MI	45			DEC	Slave: Start Of Vector
	MI	46			DEC	Read: Vector Length
OUT	MI	49			DEC	Master: Start Of Vector
	MI	51			DEC	Status Messages
	DW	26			DEC	Total Sessions
	DW	27			DEC	Acknowledgements

Parameter	**Type**	**Function**
Slave ID	Constant or MI	The IP address of the device containing the registers to be read (data source).
Slave: Start of Vector	Const, MI, ML, DW	The start of the vector of registers to be read (data source). **Note** • Check topic Slave Address Tables.
Read: Vector Length	Const, MI, ML, DW	The vector length **Note** • A MODBUS command cannot read more than 124 16-bit integers, 62 double registers, or 64 float registers at one time. In addition, 0 is not a legal length. • If, within the Slave: Start of Vector parameter, the selected register type is a **32-bit double register** (slave addresses 5100 and greater)the preset vector length must be doubled as well. If, for example: - Slave: Start of Vector parameter is set to 6300, and - You wish to preset 4 registers, for a total of 16 bytes - You must set the Preset Vector length to 8. Note that this means that, in these cases, the Preset: Vector Length parameter will always be an even number.
Master: Start of Vector	MI	This is the start of a vector of MIs that will contain the registers' values in the master (data destination).
Status Messages	MI	Shows a message number. To check status and diagnose errors, check the MODBUS Status Operands.
Total Sessions	DW	This is the number of times the master PLC will attempt to access the slave device. Note that this is a simple incremental counter. Initialize it by storing 0 into the selected DW.
Acknowledgements	DW	This is the number of times the slave device answers.

Read Float Registers (4)

Use this command to read the values of a selected group of floating point registers in a slave device and write them into a defined vector of registers in the master. Values after the decimal point are rounded to the nearest whole value. MODBUS IP Operations are located on the FBs menu.





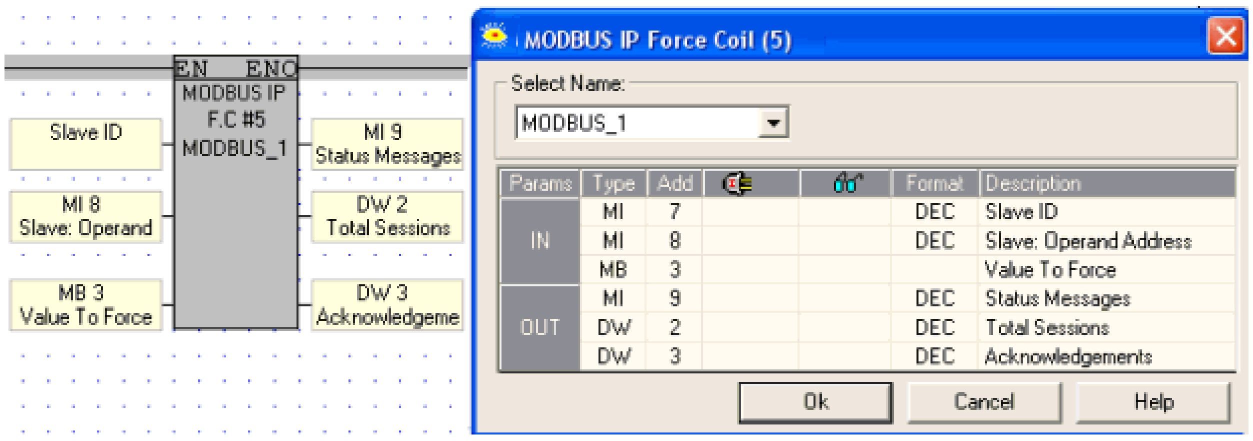
Select Name: MODBUS IP_2

Params	Type	Add			Format	Description
IN	MI	39			DEC	Slave
	MI	40			DEC	Slave: Start Of Vector
	MI	41			DEC	Read: Vector Length
OUT	MI	42			DEC	Master: Start Of Vector
	MI	43			DEC	Status Messages
	DW	24			DEC	Total Sessions
	DW	25			DEC	Acknowledgements

Parameter	Type	Function
Slave ID	Constant or MI	The IP address of the device containing the registers to be read (data source).
Slave: Start of Vector	Const, MI, ML, DW	The start of the vector of registers to be read (data source). **Note** • Check topic Slave Address Tables.
Read: Vector Length	Const, MI, ML, DW	The vector length **Note** • A MODBUS command cannot read more than 124 16-bit integers, 62 double registers, or 64 float registers at one time. In addition, 0 is not a legal length. • If, within the Slave: Start of Vector parameter, the selected register type is a **32-bit double register** (slave addresses 5100 and greater)the preset vector length must be doubled as well. If, for example: - Slave: Start of Vector parameter is set to 6300, and - You wish to preset 4 registers, for a total of 16 bytes - You must set the Preset Vector length to 8. Note that this means that, in these cases, the Preset: Vector Length parameter will always be an even number. • You can transpose 16 bits of a 32-bit double register value by turning SB 102 MODBUS Read Long ON in your program. SB 102 is OFF by default, and must be reset by the user program.
Master: Start of Vector	MI	This is the start of a vector of MIs that will contain the registers' values in the master (data destination).
Status Messages	MI	Shows a message number. To check status and diagnose errors, check the MODBUS Status Operands.
Total Sessions	DW	This is the number of times the master PLC will attempt to access the slave device. Note that this is a simple incremental counter. Initialize it by storing 0 into the selected DW.
Acknowledgements	DW	This is the number of times the slave device answers.

Force Coil (5)

Use this command to force the status of a selected coil in a slave PLC. The coil's status is forced according to the status of a selected MB in the master PLC. MODBUS IP Operations are located on the FBs menu.

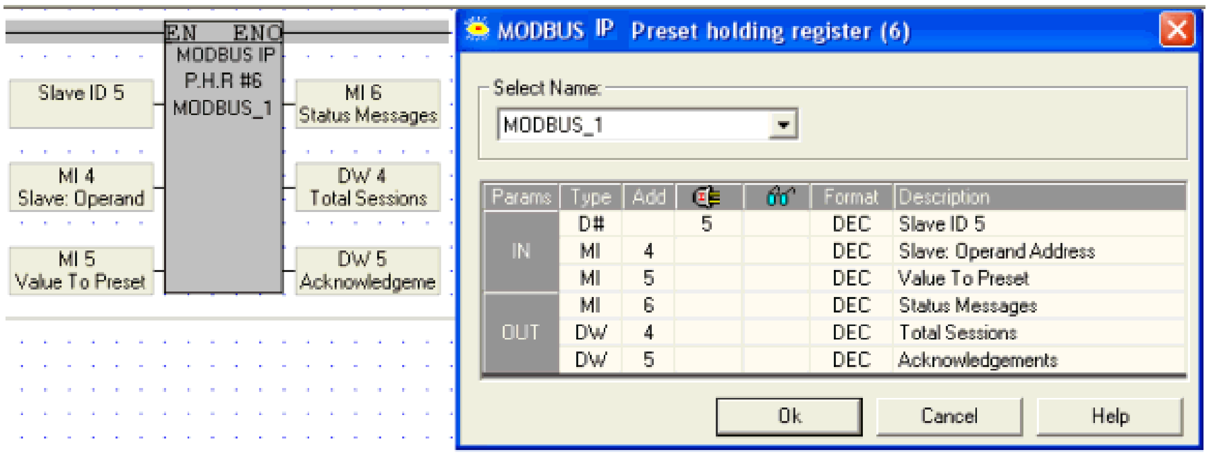


Params	Type	Add			Format	Description
IN	MI	7			DEC	Slave ID
	MI	8			DEC	Slave: Operand Address
	MB	3				Value To Force
OUT	MI	9			DEC	Status Messages
	DW	2			DEC	Total Sessions
	DW	3			DEC	Acknowledgements

Parameter	Type	Function
Slave ID	Constant or MI	The IP address of the device containing the coil to be forced (data source).
Slave Address	Const, MI, ML, DW	The address of the coil to be forced (data target). **Note** • Check topic Slave Address Tables.
Value to Force	M, SB, I, O,T	This MB is located in the master PLC; this MB contains the **status** to be forced (data source). If, for example, the status of this MB is OFF, the status of the coil in the slave will be forced to OFF. **Note** • A MODBUS command cannot read/write more than 1900 bit operands at one time. In addition, 0 is not a legal length.
Status Messages	MI	Shows a message number. To check status and diagnose errors, check the MODBUS Status Operands
Total Sessions	DW	This is the number of times the master PLC will attempt to access the slave device. Note that this is a simple incremental counter. Initialize it by storing 0 into the selected DW.
Acknowledgements	DW	This is the number of times the slave device answers.

Preset Holding Register (6)

Use this command to preset the value of a single register in a slave PLC. The value is set in a register contained in the master PLC. MODBUS IP Operations are located on the FBs menu.



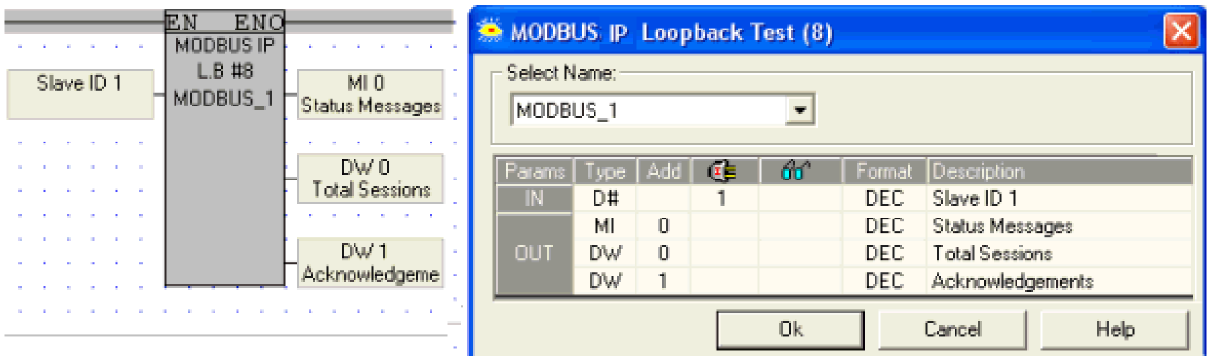
Select Name: MODBUS_1

Params	Type	Add			Format	Description
IN	D#		5		DEC	Slave ID 5
	MI	4			DEC	Slave: Operand Address
	MI	5			DEC	Value To Preset
OUT	MI	6			DEC	Status Messages
	DW	4			DEC	Total Sessions
	DW	5			DEC	Acknowledgements

Parameter	**Type**	**Function**
Slave ID	Constant or MI	The IP address of the device containing the register to be preset (target).
Slave: Operand Address	Const, MI, ML, DW	The address of the register to be preset (target). **Note** • Check topic Slave Address Tables
Value to Preset	Constant, MI, SI, ML, SL, DW, SDW or T	This is the address of the register containing the value in the master PLC (source). This value will be written into the slave's register, the register that is to be preset.
Status Messages	MI	Shows a message number. To check status and diagnose errors, check the MODBUS Status Operands
Total Sessions	DW	This is the number of times the master PLC will attempt to access the slave device. Note that this is a simple incremental counter. Initialize it by storing 0 into the selected DW.
Acknowledgements	DW	This is the number of times the slave device answers.

Loopback Test (8)

Use this command to send a test message to a slave device and receive Acknowledgements when communications are functioning properly. MODBUS IP Operations are located on the FBs menu.



MODBUS IP Loopback Test (8)

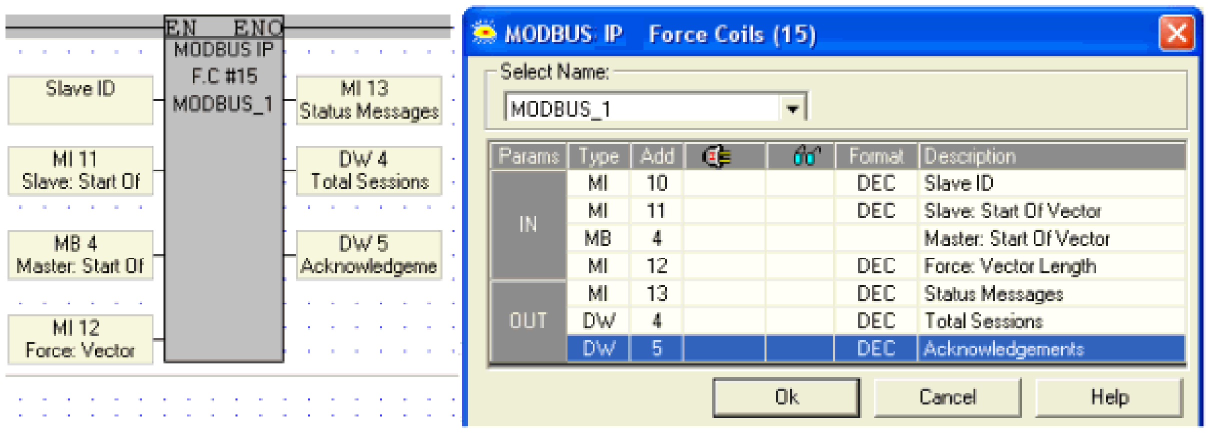
Select Name: MODBUS_1

Params	Type	Add			Format	Description
IN	D#		1		DEC	Slave ID 1
OUT	MI	0			DEC	Status Messages
	DW	0			DEC	Total Sessions
	DW	1			DEC	Acknowledgements

Parameter	Type	Function
Slave ID	Constant or MI	The IP address of the device to be checked.
Status Messages	MI	Shows a message number. To check status and diagnose errors, check the MODBUS Status Operands.
Total Sessions	DW	This is the number of times the master PLC will attempt to access the slave device. Note that this is a simple incremental counter. Initialize it by storing 0 into the selected DW.
Acknowledgements	DW	This is the number of times the slave device answers.

Force Coils (15)

Use this command to force the status of a selected group of coils in a slave PLC. The coils' status is forced according to the status of a group of MBs in the master PLC. MODBUS IP Operations are located on the FBs menu.



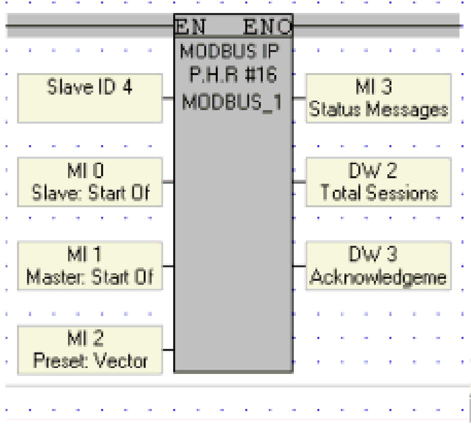
Select Name: MODBUS_1

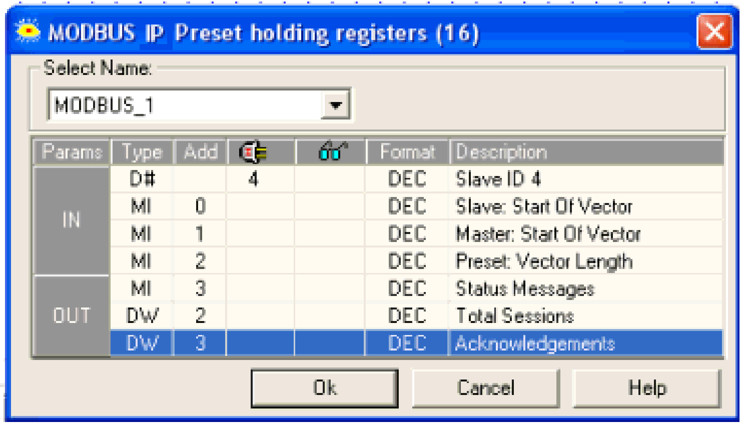
Params	Type	Add			Format	Description
IN	MI	10			DEC	Slave ID
	MI	11			DEC	Slave: Start Of Vector
	MB	4				Master: Start Of Vector
	MI	12			DEC	Force: Vector Length
OUT	MI	13			DEC	Status Messages
	DW	4			DEC	Total Sessions
	DW	5			DEC	Acknowledgements

Parameter	**Type**	**Function**
Slave ID	Constant or MI	The IP address of the slave device containing the coils to be forced (target).
Slave:Start of Vector	Const, MI, ML, DW	The start of the vector of coils to be forced (data target). **Note** • Check topic Slave Address Tables.
Master: Start of Vector	MI, SB, I, O,T	This is the start of a vector of MBs that will contain the coils' status in the master (data source).
Force: Vector Length	Constant or MI	The vector length. **Note** • A MODBUS command cannot read/write more than 1900 bit operands at one time. In addition, 0 is not a legal length.
Status Messages	MI	Shows a message number. To check status and diagnose errors, check the MODBUS Status Operands
Total Sessions	DW	This is the number of times the master PLC will attempt to access the slave device. Note that this is a simple incremental counter. Initialize it by storing 0 into the selected DW.
Acknowledgements	DW	This is the number of times the slave device answers.

Preset Holding Registers (16)

Use this command to preset the value of a group of registers in a slave PLC. The values are set in a vector of registers contained in the master PLC. MODBUS IP Operations are located on the FBs menu.





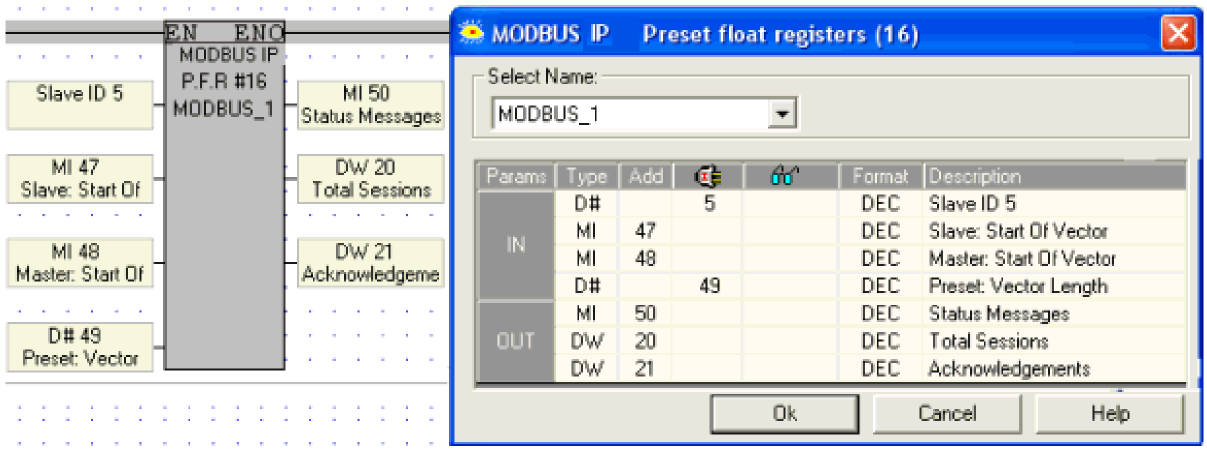
Select Name: MODBUS_1

Params	Type	Add			Format	Description
IN	D#		4		DEC	Slave ID 4
	MI	0			DEC	Slave: Start Of Vector
	MI	1			DEC	Master: Start Of Vector
	MI	2			DEC	Preset: Vector Length
OUT	MI	3			DEC	Status Messages
	DW	2			DEC	Total Sessions
	DW	3			DEC	Acknowledgements

Parameter	Type	Function
Slave ID	Constant or MI	The IP address of the device containing the registers to be preset (target).
Slave: Start of Vector	Const, MI, ML, DW	The start of the vector of registers to be preset (target). **Note** • Check topic Slave Address Tables.
Master: Start of Vector	Constant, MI, SI, ML, SL, DW, SDW or T	This is the start of a vector of MIs that will contain the registers' values in the master (data source).
Preset: Vector Length	Const, MI, ML, DW	The length of the vector of registers in both master and slave. **Note** • A MODBUS command cannot read more than 124 16-bit integers, 62 double registers, or 64 float registers at one time. In addition, 0 is not a legal length. • If, within the Slave: Start of Vector parameter, the selected register type is a **32-bit double register** (slave addresses 5100 and greater)the preset vector length must be doubled as well. If, for example: - Slave: Start of Vector parameter is set to 6300, and - You wish to preset 4 registers, for a total of 16 bytes - You must set the Preset Vector length to 8. Note that this means that, in these cases, the Preset: Vector Length parameter will always be an even number.
Status Messages	MI	Shows a message number. To check status and diagnose errors, check the MODBUS Status Operands.
Total Sessions	DW	This is the number of times the master PLC will attempt to access the slave device. Note this is a simple incremental counter. Initialize it by storing 0 into the selected DW.
Acknowledgements	DW	This is the number of times the slave device answers.

Preset Float Registers (16)

Use this command to preset the value of a group of floating point registers in a slave PLC. The values are set in a vector of registers contained in the master PLC. Values after the decimal point are rounded to the nearest whole value. MODBUS IP Operations are located on the FBs menu.



MODBUS IP Preset float registers (16)

Select Name: MODBUS_1

Params	Type	Add			Format	Description
IN	D#		5		DEC	Slave ID 5
	MI	47			DEC	Slave: Start Of Vector
	MI	48			DEC	Master: Start Of Vector
	D#		49		DEC	Preset: Vector Length
OUT	MI	50			DEC	Status Messages
	DW	20			DEC	Total Sessions
	DW	21			DEC	Acknowledgements

Parameter	**Type**	**Function**
Slave ID	Constant or MI	The IP address of the device containing the register to be preset (target).
Slave: Start of Vector	Const, MI, ML, DW	The address of the register to be preset (target). **Note** • Check topic Slave Address Tables.
Master: Start of Vector	MI, SI, ML, SL, DW, SDW or T	This is the address of the register containing the value in the master PLC (source). This value will be written into the slave's register, the register that is to be preset.
Preset: Vector Length	Const, MI, ML, DW	The length of the vector of registers in both master and slave. **Note** • A MODBUS command cannot read more than 124 16-bit integers, 62 double registers, or 64 float registers at one time. In addition, 0 is not a legal length. • If, within the Slave: Start of Vector parameter, the selected register type is a **32-bit double register** (slave addresses 5100 and greater)the preset vector length must be doubled as well. If, for example: - Slave: Start of Vector parameter is set to 6300, and - You wish to preset 4 registers, for a total of 16 bytes - You must set the Preset Vector length to 8. Note that this means that, in these cases, the Preset: Vector Length parameter will always be an even number.
Status Messages	MI	Shows a message number. To check status and diagnose errors, check the MODBUS Status Operands.
Total Sessions	DW	This is the number of times the master PLC will attempt to access the slave device. Note that this is a simple incremental counter. Initialize it by storing 0 into the selected DW.
Acknowledgements	DW	This is the number of times the slave device answers.

Read/Write Mixed Data via MODBUS

The Read/Write Mixed Data function enables you to combine Read & Write operations in a single MODBUS command, transferring data between master and slave controllers, without using standard MODBUS commands and addressing conventions.

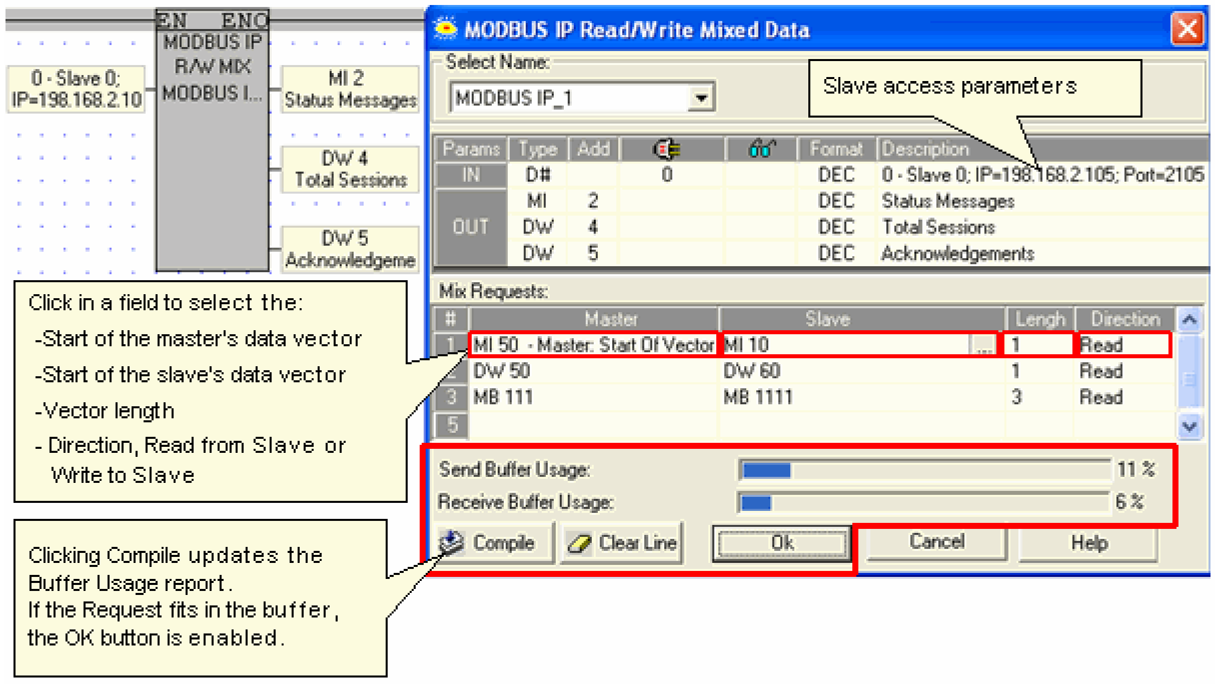
Please note that this is not a standard MODBUS function. Read/Write Mixed Data is compatible with Unitronics' Vision controllers only.

Note that this function is compatible with O/S 401 and higher, and is not supported for V120-12-xxx.

Each function can contain both Read and Write Mix Requests. Each request may be for a different data type. Your data request must include:

- Master and Slave operand addresses
- Length of vector
- Direction: Read or Write

After you add a request, the OK button is disabled. Click the Compile button to see current buffer status; if the buffer contains less than the maximum number of bytes, the OK is enabled.



Params	Type	Add			Format	Description
IN	D#		0		DEC	0 - Slave 0; IP=198.168.2.105; Port=2105
OUT	MI	2			DEC	Status Messages
	DW	4			DEC	Total Sessions
	DW	5			DEC	Acknowledgements

#	Master	Slave	Lengh	Direction
1	MI 50 - Master: Start Of Vector	MI 10	1	Read
2	DW 50	DW 60	1	Read
3	MB 111	MB 1111	3	Read
5				

Click in a field to select the:
-Start of the master's data vector
-Start of the slave's data vector
-Vector length
- Direction, Read from Slave or Write to Slave

Clicking Compile updates the Buffer Usage report.
If the Request fits in the buffer, the OK button is enabled.

Read Write Limitations

Only the following data types may be used in Mixed Data requests: MI, ML, DW, MB, I and O.

Registers: may only be read/written to the same data type.

Booleans: Inputs cannot be written to.

Booleans, Read Write		
I	→	MB, O
O	↔	MB, O
MB	↔	MB, O

Registers, Read Write		
MI	↔	MI
ML	↔	ML
DW	↔	DW

Send / Receive Buffers

The function uses two buffers, Send and Receive. Each buffer can contain a maximum of 500 bytes.

Read/Write from Data Tables

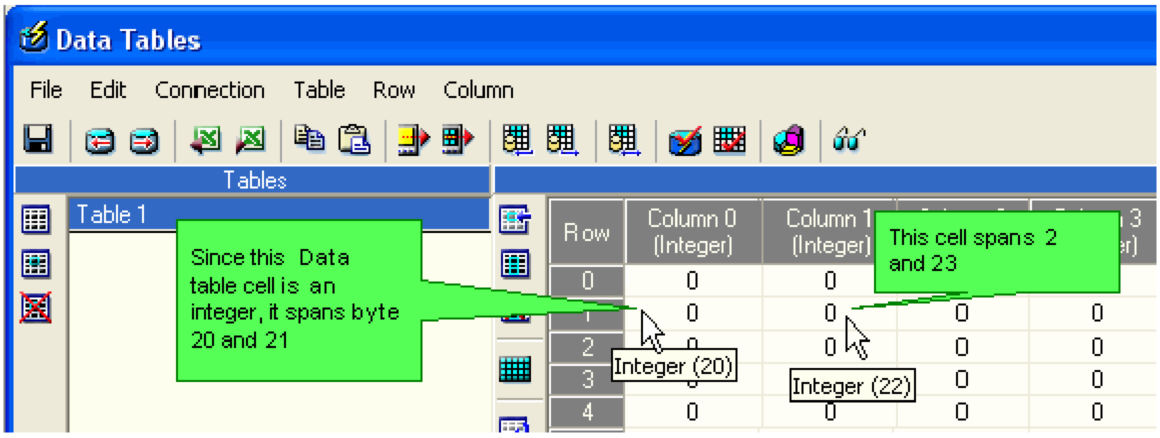
Use these commands to access the bytes in Vision data tables **without** reference to table structure.

Please note that this is not a standard MODBUS function. Read/Write from Data Tables is compatible with Unitronics' Vision PLC data tables only.

To determine the byte number of a data table cell, hold the cursor over the data table cell. A Tooltip opens, displaying the byte number.

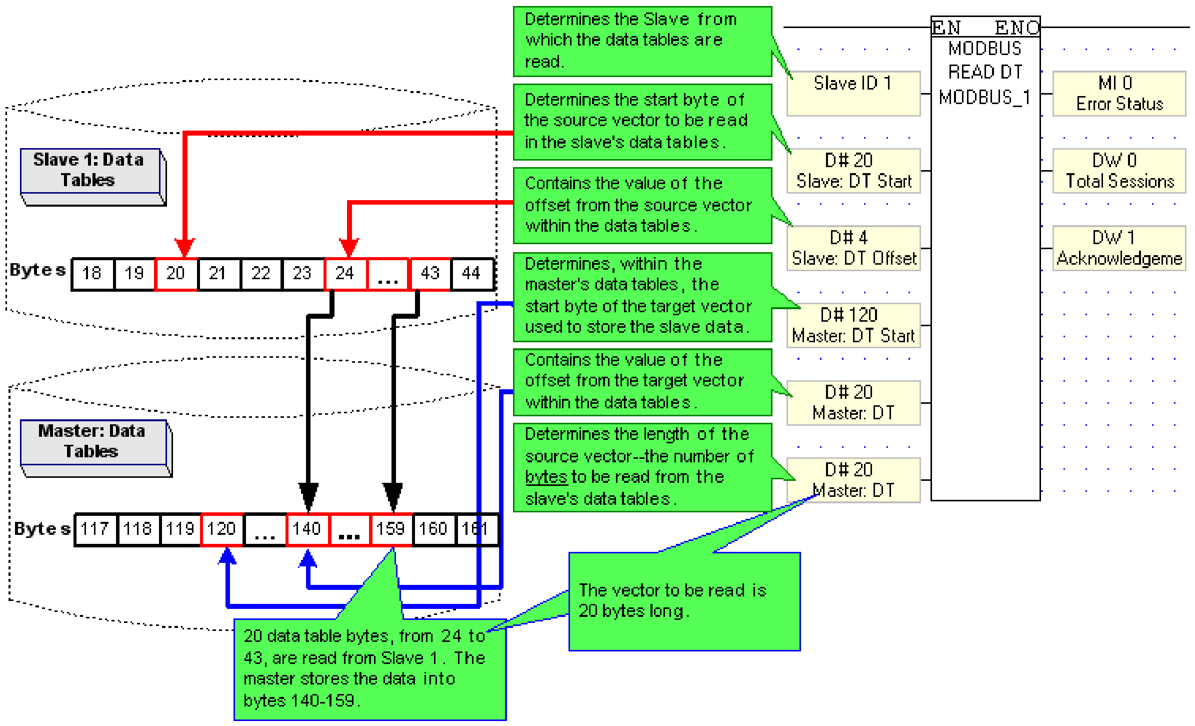
Note • A MODBUS command cannot read/write more than 242 DT bytes at one time.

In addition, 0 is not a legal length.



Read from Data Table

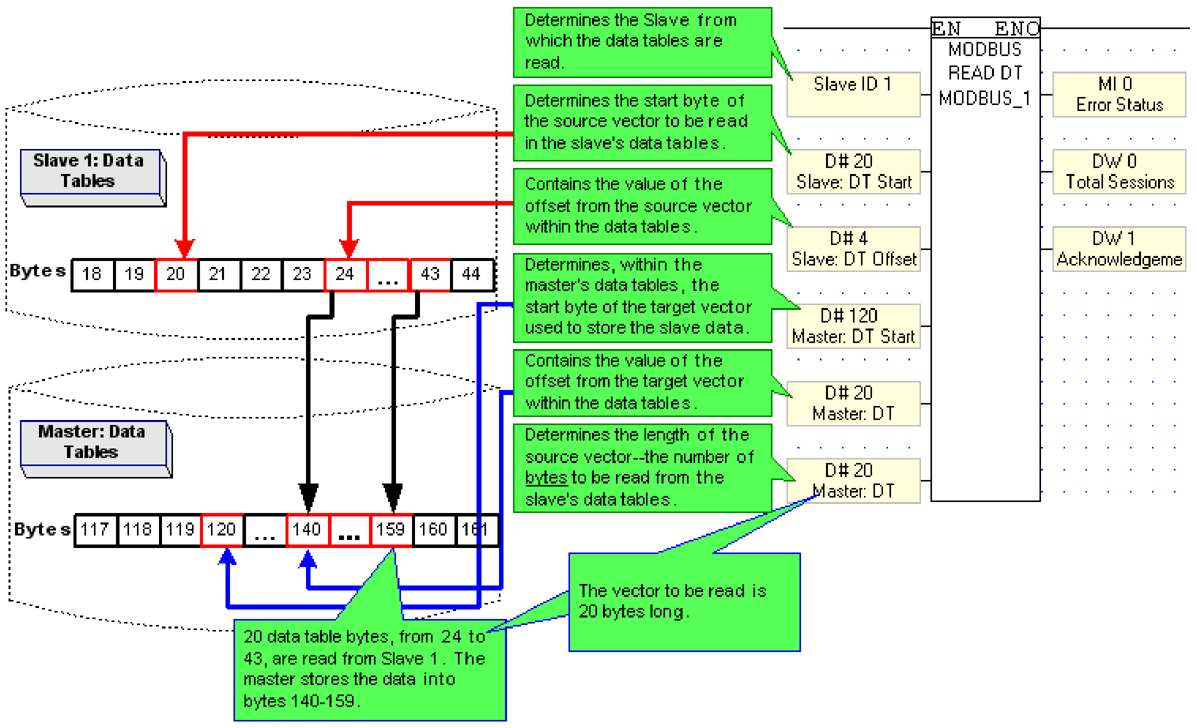
Below, a MODBUS master reads data tables in Slave ID 1. Bytes 24-43 are read from Slave 1 into bytes 140-159 in the master's data tables.


Determines the Slave from which the data tables are read.
Determines the start byte of the source vector to be read in the slave's data tables.
Contains the value of the offset from the source vector within the data tables.
Determines, within the master's data tables, the start byte of the target vector used to store the slave data.
Contains the value of the offset from the target vector within the data tables.
Determines the length of the source vector--the number of bytes to be read from the slave's data tables.
The vector to be read is 20 bytes long.
20 data table bytes, from 24 to 43, are read from Slave 1. The master stores the data into bytes 140-159.
Slave 1: Data Tables
Bytes
18
19
20
21
22
23
24
...
43
44
Master: Data Tables
Bytes
117
118
119
120
...
140
...
159
160
161
EN
ENO
MODBUS READ DT
MODBUS_1
Slave ID 1
D# 20 Slave: DT Start
D# 4 Slave: DT Offset
D# 120 Master: DT Start
D# 20 Master: DT
D# 20 Master: DT
MI 0 Error Status
DW 0 Total Sessions
DW 1 Acknowledgeme

Parameter	Type	Function
Slave ID	Constant or MI	The ID of the slave device containing the coils to be read (data source).
Slave: DT Start of Vector	Const, MI, ML, DW	The start of the vector of bytes to be read (data source).
Slave: DT Offset in Vector	Const, MI, ML, DW	Offset from the Slave: DT Start of Vector
Master: DT Start of Vector	Const, MI, ML, DW	This is the start of a vector of bytes that will contain the data read from the slave.
Master: DT Offset in Vector	Const, MI, ML, DW	Offset from the Master: DT Start of Vector
Read: DT Vector Length	Constant or MI	The vector length. **Note •** A MODBUS command cannot read/write more than 242 DT bytes at one time. In addition, 0 is not a legal length.
Status Messages	MI	Shows a message number. To check status and diagnose errors, check the MODBUS Status Messages.
Total Sessions	DW	This is the number of times the master PLC will attempt to access the slave device. Note that this is a simple incremental counter. Initialize it by storing 0 into the selected DW.
Acknowledgements	DW	This is the number of times the slave device answers.

Write to Data Table

Below, a MODBUS master writes to data tables in Slave ID 1. Bytes 140-159 are written from the master into bytes 24-43 in the slave's data tables.


Determines the Slave from which the data tables are read.
Determines the start byte of the source vector to be read in the slave's data tables.
Contains the value of the offset from the source vector within the data tables.
Determines, within the master's data tables, the start byte of the target vector used to store the slave data.
Contains the value of the offset from the target vector within the data tables.
Determines the length of the source vector--the number of bytes to be read from the slave's data tables.
The vector to be read is 20 bytes long.
20 data table bytes, from 24 to 43, are read from Slave 1. The master stores the data into bytes 140-159.
Slave 1: Data Tables
Master: Data Tables
Bytes
EN ENO
MODBUS READ DT MODBUS_1
Slave ID 1
D# 20 Slave: DT Start
D# 4 Slave: DT Offset
D# 120 Master: DT Start
D# 20 Master: DT
D# 20 Master: DT
MI 0 Error Status
DW 0 Total Sessions
DW 1 Acknowledgeme

Parameter	Type	Function
Slave ID	Constant or MI	The ID of the slave device to which the data will be written (data target).
Slave: DT Start of Vector	Const, MI, ML, DW	The start of the vector of bytes to be written into (data target).
Slave: DT Offset in Vector	Const, MI, ML, DW	Offset from the Slave: DT Start of Vector
Master: DT Start of Vector	Const, MI, ML, DW	This is the start of a vector of bytes, in the master, that will contain the data to be written to the slave (data source)
Master: DT Offset in Vector	Const, MI, ML, DW	Offset from the Master: DT Start of Vector
Read: DT Vector Length	Constant or MI	The vector length. **Note** • A MODBUS command cannot read/write more than 242 DT bytes at one time. In addition, 0 is not a legal length.
Status Messages	MI	Shows a message number. To check status and diagnose errors, check the MODBUS Status Messages.
Total Sessions	DW	This is the number of times the master PLC will attempt to access the slave device. Note that this is a simple incremental counter. Initialize it by storing 0 into the selected DW.
Acknowledgements	DW	This is the number of times the slave device answers.

Configuring a MODBUS slave device

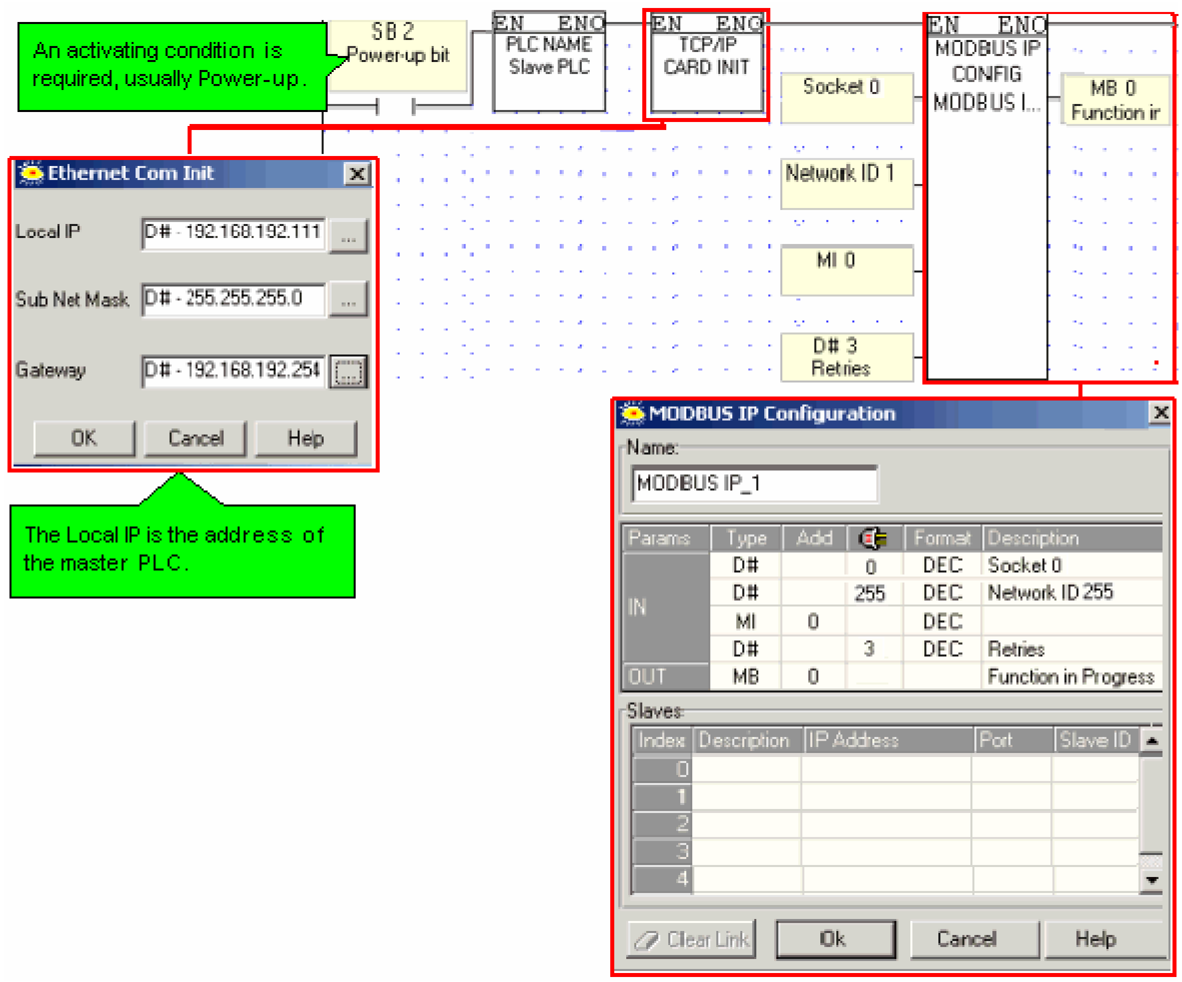
The Ladder section below shows what elements are necessary to enable a master device to read from a slave. Note that the MODBUS Scan operation should **not** be performed during the initial program scan.

Note that you must use a condition (RLO) to activate the MODBUS Configuration.

The slave PLC Ladder application must include the elements shown below.

Step 1: Initializing the Ethernet card and configuring MODBUS

Note that all slave devices must be assigned ID number 255.


An activating condition is required, usually Power-up.
SB 2 Power-up bit
EN ENO PLC NAME Slave PLC
EN ENO TCP/IP CARD INIT
Socket 0
Network ID 1
MI 0
D# 3 Retries
EN ENO MODBUS IP CONFIG MODBUS I...
MB 0 Function ir
The Local IP is the address of the master PLC.

Ethernet Com Init

Local IP	D# - 192.168.192.111
Sub Net Mask	D# - 255.255.255.0
Gateway	D# - 192.168.192.254

OK Cancel Help

MODBUS IP Configuration

Name: MODBUS IP_1

Params	Type	Add		Format	Description
IN	D#		0	DEC	Socket 0
	D#		255	DEC	Network ID 255
	MI	0		DEC	
	D#		3	DEC	Retries
OUT	MB	0			Function in Progress

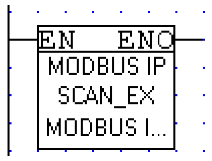
Slaves:

Index	Description	IP Address	Port	Slave ID
0				
1				
2				
3				
4				

Clear Link Ok Cancel Help

Step 2: Scan

To enable the master PLC to access the slave, include a MODBUS Scan FB in the slave's application.



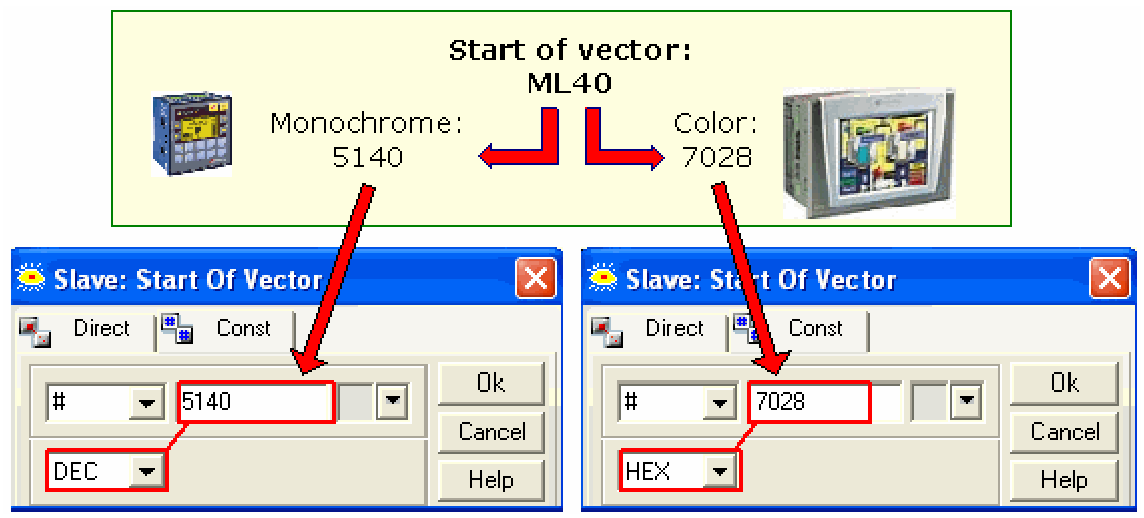
Slave Addressing

Monochrome / Color PLCs

The memory structure of monochrome screen PLCs is different from that of color screen PLCs. This is why each series has its own slave addressing scheme.

Note • The slave addresses given for the monochrome series are decimal values.

The slave addresses given for the color series are hexadecimal values.



Slave Addresses

Standard Vision Division

Coils		MODBUS Command Number	
Pointer Value From:	Operand type	Read	Write
0000	MB 0-2999	#01 Read Coils	#15 Force Coils
3000	SB		#15 Force Coils
4000	I (read-only)		Read-only
5000	O		#15 Force Coils
6000	T (read-only)		Read-only
7000	C (read-only)		Read-only
8000	MB 3000-4095		

Note • Note that in order to access MBs 3000-4095, you address as follows: to access MB 3012, request slave address 8012.

Registers			MODBUS Command Number	
Pointer Value From:	Operand type	Register size	Read	Write
0000	MI	16 bit	# 03 Read Holding Registers	# 16 Preset Holding Registers
4000	SI	16 bit		
5100	ML	32 bit		
6100	SL	32 bit		
6300	MDW	32 bit		
6700	SDW	32 bit		
6900	Timer preset	32 bit		
7300	Timer current	32 bit		
7700	MF	32 bit		
7800	Counter Preset	16 bit		
7900	Counter Current	16 bit		

Enhanced Vision Series

Coils		MODBUS Command Number	
Pointer Value From (hex):	Operand type	Read	Write
0000h	MB 0	#01 Read Coils	#15 Force Coils
3000h	XB		#15 Force Coils
4000h	O		#15 Force Coils

5000h	SB		Read-only
6000h	I (read-only)		#15 Force Coils
7000h	T (read-only)		Read-only
8000h	C (read-only)		Read-only

Registers			MODBUS Command Number	
Pointer Value From (hex):	Operand type	Register size	Read	Write
0000h	MI	16 bit	# 03 Read Holding Registers	# 16 Preset Holding Registers
3000h	XI			
9000h	SI	16 bit		
5000h	XL			
6000h	XDW			
7000h	ML	32 bit		
A000h	SL	32 bit		
8000h	MDW	32 bit		
B000h	SDW	32 bit		
C000h	Timer preset	32 bit		
D000h	Timer current	32 bit		
4000h	MF	32 bit		
E000h	Counter Preset	16 bit		
F000h	Counter Current	16 bit		

Examples

The examples below show that:

- MODBUS addressing systems start at 1.
- Vision addressing starts at 0.

Bit Operands

Read a 10-bit vector of inputs in a slave Vision controller, starting at Input 20, via Read Coils (MODBUS COMMAND #1)

- Vision PLC as the MODBUS master to Monochrome PLC

In VisiLogic's Read Coils FB, set the Slave: Start of Vector parameter to 4020 (DEC), and the Read: Vector Length parameter to 10. Within the slave Vision controller, VisiLogic will read I 20 - I 29.

- Vision PLC as the MODBUS master to Color PLC

In VisiLogic's Read Coils FB, set the Slave: Start of Vector parameter to 6014h (HEX), and the Read: Vector Length parameter to 10. Within the slave Vision controller, VisiLogic will read I 20 - I 29.

- SCADA as the MODBUS master to Monochrome PLC

In the SCADA application, set the Slave: Start of Vector parameter to 34021(30001 + 4000 + 20), and the Read: Vector Length to 10, enabling the Master device to read I 20 - I 29 within the slave Vision controller.

- **SCADA as the MODBUS master to Color PLC**

Convert the HEX address to DEC
In the SCADA application, set the Slave: Start of Vector parameter to 54597(24576 (6000h) + 20), and the Read: Vector Length to 10, enabling the Master device to read I 20 - I 29 within the slave Vision controller.

Write a 3-bit vector of outputs in a slave Vision controller, starting at Output 8, via Force Coils (MODBUS COMMAND #15)

- Vision PLC as the MODBUS master to Monochrome PLC

In VisiLogic's Force Coils FB, set the Slave: Start of Vector parameter to 5008, and the Force: Vector Length parameter to 3. Within the slave Vision controller, the master will force the status of O 8 - O 10.

- Vision PLC as the MODBUS master to Color PLC

In VisiLogic's Force Coils FB, set the Slave: Start of Vector parameter to 4008h (HEX), and the Force: Vector Length parameter to 3. Within the slave Vision controller, the master will force the status of O 8 - O 10.

- SCADA as the MODBUS master to Monochrome PLC

In the SCADA application, set the Slave: Start of Vector parameter to 35009 (30001 + 5000 + 8) and the Force: Vector Length parameter to 3, enabling the Master device to write to O 8 - O 10 within the slave Vision controller.

- **SCADA as the MODBUS master to Color PLC**

Convert the HEX address to DEC.
In the SCADA application, set the Slave: Start of Vector parameter to 46393 (30001 + 16384(4000h) + 8) and the Force: Vector Length parameter to 3,

enabling the Master device to write to O 8 - O 10 within the slave Vision controller.

Registers

Read a 9-register long vector of **16-bit integers** in a slave Vision controller, starting at MI 32, via Read Holding Registers (MODBUS COMMAND #03)

- Vision PLC as the MODBUS master

In VisiLogic's Read Holding Registers FB, set the Slave: Start of Vector parameter to 32, and the Read: Vector Length parameter to 9. Within the slave Vision controller, VisiLogic will read MI 32 - MI 41.

- SCADA as the MODBUS master

In the SCADA application, set the Slave: Start of Vector parameter to 40033 (40001 + 0000 + 3), and the Read: Vector Length parameter to 9, enabling the Master device to read MI 32 - MI 41 within the slave Vision controller.

Note • If, within the Slave: Start of Vector parameter, the selected register type is a **32-bit double register** (slave addresses 5100 and greater) the preset vector length must be **doubled** as well.
If, for example in the VisiLogic Preset Holding Registers FB:

- Slave: Start of Vector parameter is set to 6300, and
- You wish to preset 4 registers, for a total of 16 bytes
- You must set the Preset Vector length to 8.

Note that this means that, in these cases, the **Preset: Vector Length** parameter will always be an even number.

Read a 9-register long vector of **32 -bit integers** in a slave Vision controller, starting at SL 32, via Preset Holding Registers (MODBUS COMMAND #16)

- Vision PLC as the MODBUS master

In VisiLogic's Preset Holding Registers FB, set the Slave: Preset Vector parameter to 6132, and the Read: Vector Length parameter to 18 (2x9, in order to fit the 32-bit SL registers). Within the slave Vision controller, VisiLogic will read SL 32 - SL 41.

- SCADA as the MODBUS master

In the SCADA application, set the Slave: Start of Vector parameter to 406133, and the Read: Vector Length parameter to 18, enabling the Master device to read SL 32 - SL 41 within the slave Vision controller.

Write a 6-register long vector of **16-bit integers** in a slave Vision controller, starting at MI 32, via Preset Holding Registers (MODBUS COMMAND #16)

- Vision PLC as the MODBUS master

In VisiLogic's Preset Holding Registers FB, set the Slave: Start of Vector parameter to 32, and the Preset: Vector Length parameter to 6. Within the slave Vision controller, VisiLogic will write to MI 32 - MI 37.

- SCADA as the MODBUS master

In the SCADA application, set the Slave: Start of Vector parameter to 40033, and the Read: Vector Length parameter to 6, enabling the Master device to write to MI 32 - MI 37 within the slave Vision controller.

MODBUS Status Operands

All of the Status operands linked to MODBUS FBs should be assigned Power-up Values; bits should be reset, and registers initialized to 0.

MODBUS: Configuration FB Status Operand

All MODBUS operations run through a MODBUS configuration placed in the master device's program.

Function in Progress Shows status of master's **MODBUS Configuration**	**MB**	**Turns ON when:** A master Vision initiates MODBUS communication. Remains ON during the MODBUS session.	**Turns OFF when** The MODBUS: Configuration is activated. An answer is received from a slave. The TimeOut defined in the **Configuration** is exceeded. Certain Status Messages are given

MODBUS Operation Status Operands

When you place MODBUS operations in your application (Force, Read, Preset, and Loopback commands), you link the operands below. These show the status of MODBUS sessions.

Status Messages Shows status of master's data requests and the replies the master receives from the slaves	**MI**	Automatically initialized to 0 when MODBUS operation is activated. Updated at the end of each attempt to communicate via MODBUS. Indicates status of **MODBUS** communications, according to the table below. Note that the current value always shows the most **recent** status.

#	**Status Message**
0	Status OK
1	Unknown Command Number This is received from the slave device**.**
2	Illegal Data Address Master: an invalid address is found by the master before a data request is sent to a slave. This may result, for example, when an MI is used to provide vector length. Slave: The slave notifies the master that the data request command includes invalid addresses. Slave--ScanEX: When ScanEX receives an input parameter in the 32-bit range (for example, 5100{ML}), it automatically takes double-register values. If, for example, ScanEX receives a Read Register(6) request for **5100**, it returns the values in 5100 **and** 5101. If, however, ScanEX receives Read Register(6) request for **5101**, it returns Error #2-- since 5101 provides the 'high' bytes of the 32-bit register, it is not a legal address.
3	**Slave to Master: Illegal Data Type Quantity** Number of operands requested by user exceeds the maximum **Note •** A MODBUS command cannot read more than 124 16-bit integers, 62 double registers, 62 float registers, or 1900 bit operands at one time. In addition, 0 is not a legal vector length.
4	**Master--Time Out** This occurs if the master has been waiting for a slave response for an amount of time exceeding the Time-out set in the Configuration. If this error occurs, check **Time Out, Baud Rate, and Distance**
5	**No Communication** The MODBUS session cannot be established.
Note • Messages 4 & 5. **TimeOut** and **Number of Retries** are defined in the **Configuration**. A Retry is an attempt to establish a MODBUS session. If, for example, TimeOut is defined as 2 seconds, and number of Retries as 3: - the controller will try to establish the session once, and will continue to try for 2 seconds. - If the first attempt fails, the **Status Message value will be 4**, Master TimeOut. -The controller will try twice more, for a total of 3 retries over 6 seconds. - If all attempts fail, the **Status Message value will be 5**. -If any attempt succeeds, the Status Message will be 0.	
* 6	Master-slave data incorrectly synchronized

* 7	Master-slave data incorrectly synchronized
8	**Master to application: Illegal Data Type Quantity** Number of operands requested by user exceeds the maximum permitted for that FB operation in the master. **Note** • A MODBUS command cannot read more than 124 16-bit integers, 62 double registers, 62 float registers, or 1900 bit operands at one time. In addition, 0 is not a legal vector length.
9	Slave ID =0 An attempt does to communicate with Slave ID 0.
10	Incorrect CRC
* 11	Master-slave data incorrectly synchronized

***** Messages 6, 7, and 11mean that the master has found incompatible elements in the data sent between master and slave.

Total Sessions	DW	This is the total number of times the master PLC attempts to access the slave device, whether the attempt is successful or not, including Retries. Note that this is a simple incremental counter. This must be initialized by the user, by storing 0 into the selected DW.
Acknowledgements	DW	This is the number of times the slave device answers. This must be initialized by the user, by storing 0 into the selected DW.

Time Out, Baud Rate, and Distance

The default Time Out set in the Configuration is one second. In certain cases, the length of the network may cause this to be exceeded. You can calculate a Time Out value by using the following formula, and use it in the Configuration.

$$\text{Time Out} = \frac{(\text{TX bytes} + \text{RX bytes} + 7) \times 1000}{\text{baud rate} / (\text{data bits} + \text{start bits} + \text{stop bits}) * 10\text{ms}}$$

Note •	To automatically reconnect a lost Ethernet connection, turn SB 168 Automatically reconnect (keep alive) ON at power-up.
•	To enable PLC to disconnect when the Ethernet connection is inactive for a period of time, assign timeout values in SIs 103 -106
•	To enable a PLC attempt to reconnect when there is no communication from the connected device for a period of time, assign timeout values in SIs 107 -110

SMS Messaging

SMS Messaging Overview

To enable a controller to use SMS messaging, connect it to a modem that supports connection to a cellular network with SMS messaging service.
SMS messaging operations are located on the FBs menu.

To send or receive SMS messages:

1. Initialize one of the controller's communications ports using a COM Init FB.

Note	
•	Communications cannot flow through the port during initialization. To avoid conflicts in your program, use the Modem Initialization Status SBs.
•	Before using a modem, prepare it as described in the help topic PLC-side Modems.

Place an SMS Configuration FB into your application. Link it to the port initialized via the COM Init FB. The SMS Configuration will use this port to receive and send messages.

The SMS Configuration contains a list of SMS messages and phone numbers.

Note • COM Init and SMS Configuration must appear in the application and be activated before any SMS operations are activated. If this is not done, the application will not work.

To enable the controller to receive messages, place an SMS Scan FB in your application and link it to a Configuration. When activated, this causes the controller to scan the GSM modem's SIM card for incoming SMS messages

To enable the controller to send SMS messages, place an SMS Send FB in your application and link it to a Configuration. You can then either send one of the messages in that Configuration or create one for that specific Send FB.

In either case, the port defined in the Configuration will be used to send the message.

FB Operations

SMS: Configuration

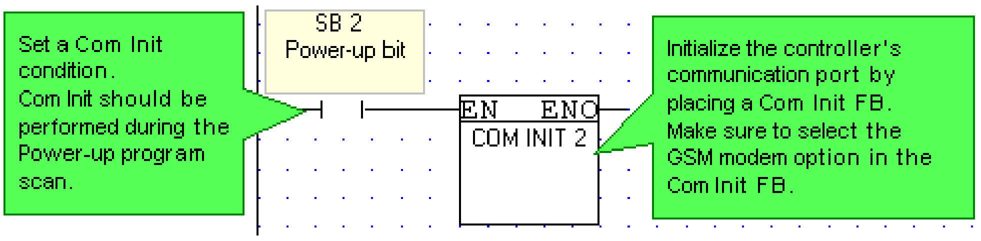
SMS: Scan

SMS: Send

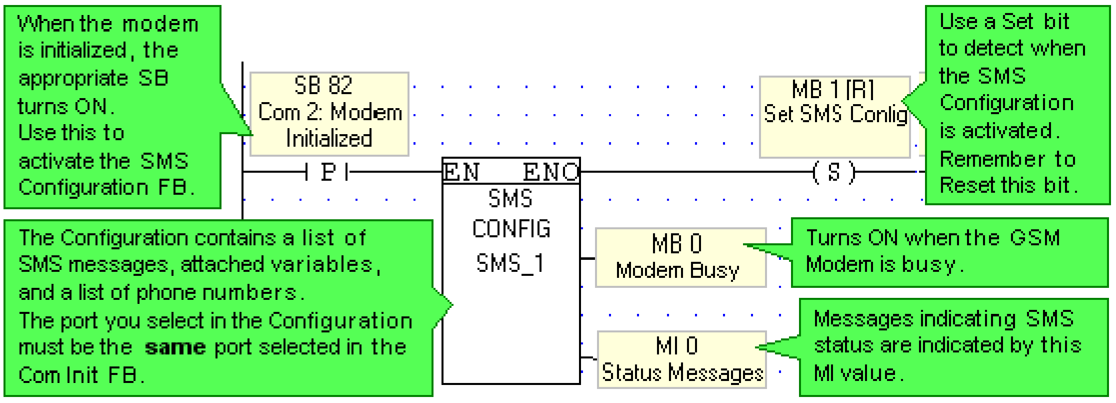
SMS: Check GSM Signal Quality

How to Configure, Receive, and Send Messages

1. Initialize a COM Port. The initialization status of each port is indicated in Modem Initialization Status SBs. These SBs can be used to activate an SMS configuration.


Set a Com Init condition. Com Init should be performed during the Power-up program scan.
SB 2 Power-up bit
EN ENO
COM INIT 2
Initialize the controller's communication port by placing a Com Init FB. Make sure to select the GSM modem option in the Com Init FB.

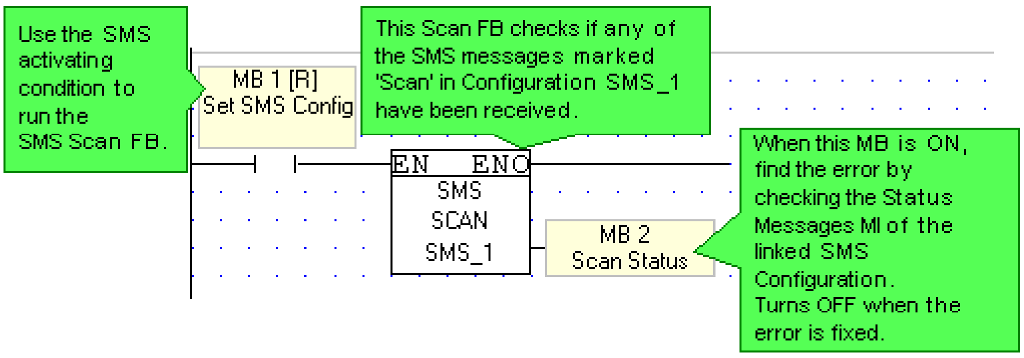
2. Select SMS Configuration from the FBs menu and place the function in your application.


When the modem is initialized, the appropriate SB turns ON. Use this to activate the SMS Configuration FB.
SB 82 Com 2: Modem Initialized
P
EN ENO
SMS CONFIG SMS_1
MB 1 [R] Set SMS Config
S
Use a Set bit to detect when the SMS Configuration is activated. Remember to Reset this bit.
The Configuration contains a list of SMS messages, attached variables, and a list of phone numbers. The port you select in the Configuration must be the same port selected in the Com Init FB.
MB 0 Modem Busy
Turns ON when the GSM Modem is busy.
MI 0 Status Messages
Messages indicating SMS status are indicated by this MI value.

Note •	The condition that activates the SMS Configuration must turn ON for a single program scan (positive transition recommended).
•	Once an SMS Configuration is activated, it can process SMS messages. However, messages are sent or received **only** when the MB linked to Modem Busy (GSM) is **OFF**. This MB turns ON when the modem is processing a message, or is communicating data to another application such as VisiLogic.
•	The appropriate Modem: Initialized SBs [80 (COM 1), SB 82 (COM 2), SB 84 (COM 3)] must turn ON before activating an SMS Configuration using that COM port; the SBs should be used as an activating condition.

The SMS configuration must be scanned during every program cycle--after the Configuration is activated. One way to ensure this is by placing the configuration in the first subroutine of the main module.

3. To receive messages, place a Scan FB in your application.


Use the SMS activating condition to run the SMS Scan FB.
MB 1 [R] Set SMS Config
This Scan FB checks if any of the SMS messages marked 'Scan' in Configuration SMS_1 have been received.
EN ENO
SMS SCAN SMS_1
MB 2 Scan Status
When this MB is ON, find the error by checking the Status Messages MI of the linked SMS Configuration. Turns OFF when the error is fixed.

Note •	An SMS Scan generally uses a direct contact as an activating condition. This enables the configuration to continually 'listen' for incoming messages. When the controller registers that a

Scan has been activated, the MB linked to Modem Busy (GSM) turns ON, turning OFF only after the Scan is complete.

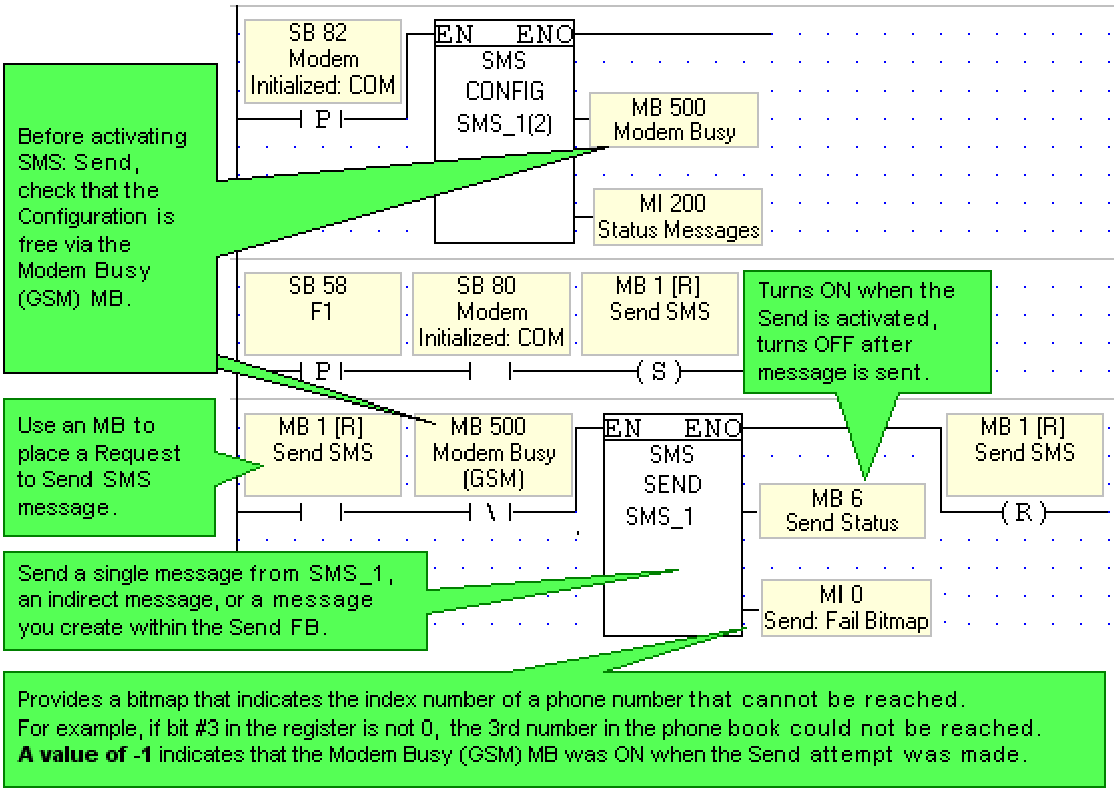
TIP

You can also write incoming messages to an operand vector, including messages that are not in the Configuration.

1. Turn SB 199.
2. Enter the Start of Vector address (MI) in SI 199. You can write to an XI vector by entering a negative value.

If SB 199 is ON, SB 198 will be SET when an SMS message is received, and the message will be stored in operand vector. The length of the message, in bytes, will be stored in SI 198.

4. To send messages, place a Send FB in your application.



Before activating SMS: Send, check that the Configuration is free via the Modem Busy (GSM) MB.

Turns ON when the Send is activated, turns OFF after message is sent.

Use an MB to place a Request to Send SMS message.

Send a single message from SMS_1, an indirect message, or a message you create within the Send FB.

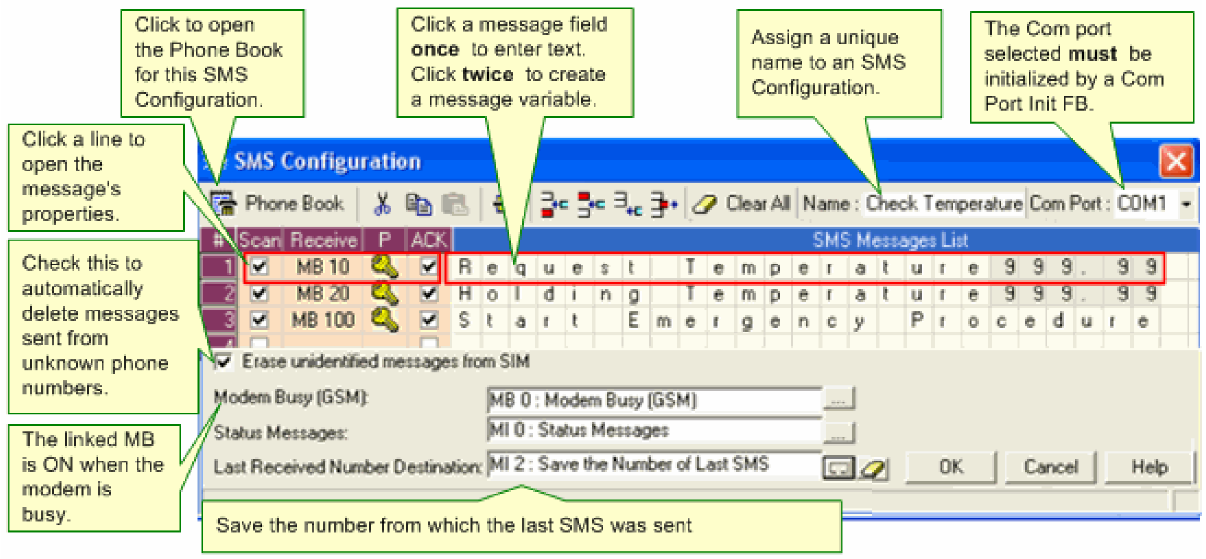
Provides a bitmap that indicates the index number of a phone number that cannot be reached. For example, if bit #3 in the register is not 0, the 3rd number in the phone book could not be reached. **A value of -1** indicates that the Modem Busy (GSM) MB was ON when the Send attempt was made.

Note

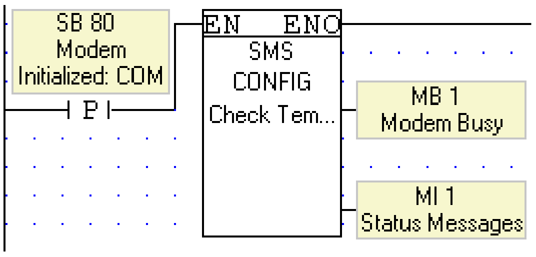
- When the positive transition contact used to activate the Send FB rises, the MB linked to Modem Busy (GSM) turns ON. The rest of the logic in the net is processed whether or not the Send operation is completed.
- Operands linked to SMS FBs, such as Modem Busy (GSM), should be assigned Power-up Values; bits should be reset, and registers initialized to 0.
- When an SMS message is sent, the Modem Busy (GSM) MB remains ON until the message has cleared the SIM card.
- Be careful when using positive transition contacts to activate a Send operation. When a Send operation is in progress, the modem is busy. If another Send operation is activated while the modem is busy, the second Send operation will not be activated and the message will **not** be sent--even after the modem is no longer busy.

SMS: Configuration

An SMS Configuration contains a list of SMS messages with attached variables and a phone book that is unique to that configuration. SMS Configuration is also where you define Message Properties.


Click to open the Phone Book for this SMS Configuration.
Click a message field **once** to enter text. Click **twice** to create a message variable.
Assign a unique name to an SMS Configuration.
The Com port selected **must** be initialized by a Com Port Init FB.
Click a line to open the message's properties.
Check this to automatically delete messages sent from unknown phone numbers.
The linked MB is ON when the modem is busy.
Save the number from which the last SMS was sent
SMS Configuration
Phone Book
Clear All
Name : Check Temperature
Com Port : COM1
#
Scan
Receive
P
ACK
SMS Messages List
1 MB 10 Request Temperature 999.99
2 MB 20 Holding Temperature 999.99
3 MB 100 Start Emergency Procedure
Erase unidentified messages from SIM
Modem Busy (GSM): MB 0 : Modem Busy (GSM)
Status Messages: MI 0 : Status Messages
Last Received Number Destination: MI 2 : Save the Number of Last SMS
OK
Cancel
Help

Note • The appropriate Modem: Initialized SB [80 (COM 1), SB 82 (COM 2), SB 84 (COM 3)] must turn ON before the application can activate an SMS Config FB using that COM port.
A Modem Initialized SB should be used as an activating condition.

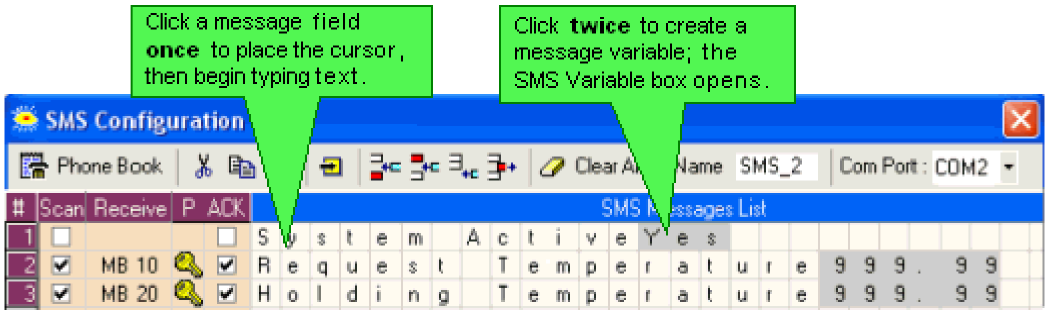


Creating SMS Messages & Variables

A single SMS message can contain both fixed text and up to 10 variables. A message containing only English characters may contain up to 160 characters. A message containing non-English characters may contain up to 70 characters.

Note • When you create messages in a Configuration, VisiLogic does not allow you to begin a message with a fixed text character that is not a numeral (0-9) or a number sign (#). These limitations do not apply to messages that you compose in an SMS Send FB, or to Indirect Messages. These may begin with variables or with any alphanumeric character.

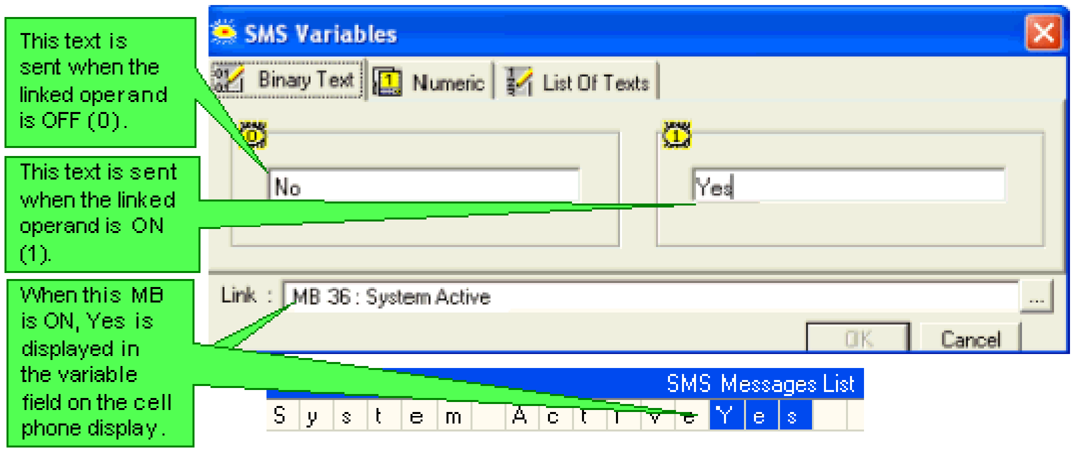
• Although the PLC can send Binary Text, Numeric, and List of Texts variables, it can only receive **Numeric variables**..


Click a message field **once** to place the cursor, then begin typing text.
Click **twice** to create a message variable; the SMS Variable box opens.
SMS Configuration
Phone Book
Clear All
Name SMS_2
Com Port : COM2
SMS Messages List

#	Scan	Receive	P	ACK	SMS Messages List
1	☐			☐	System Active Yes
2	☑	MB 10		☑	Request Temperature 999.99
3	☑	MB 20		☑	Holding Temperature 999.99

Binary Variable

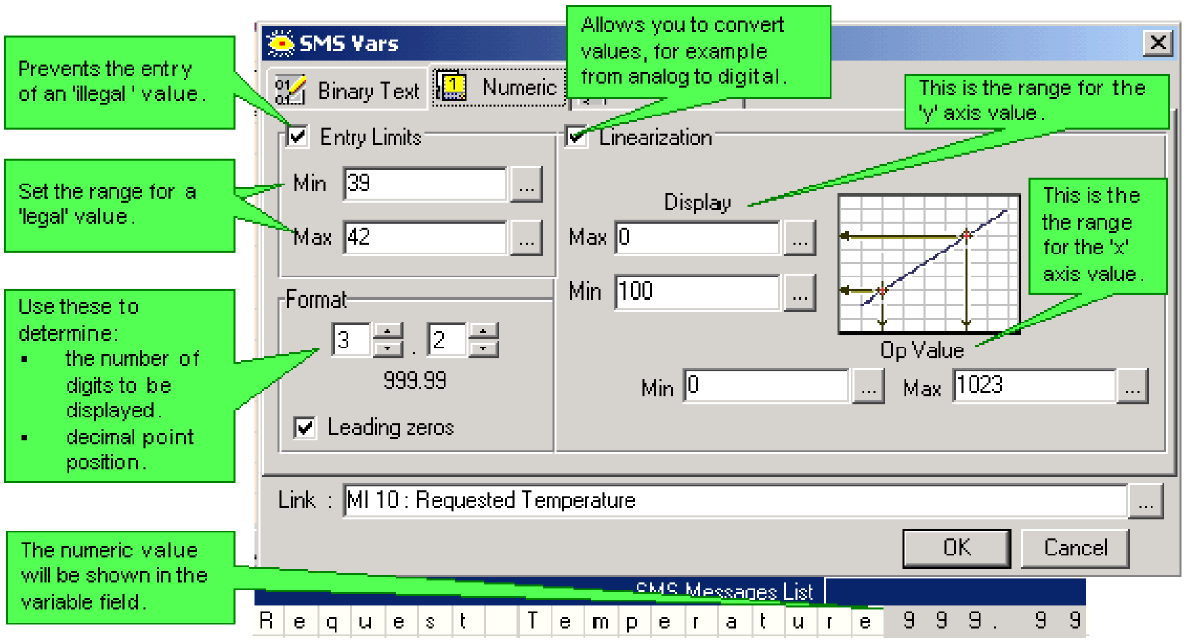
This type of variable displays different text in the SMS variable field according to the status of a bit operand.


This text is sent when the linked operand is OFF (0).
This text is sent when the linked operand is ON (1).
When this MB is ON, Yes is displayed in the variable field on the cell phone display.
SMS Variables
Binary Text
Numeric
List Of Texts
No
Yes
Link : MB 36 : System Active
OK
Cancel
SMS Messages List
System Active Yes

Number Variable

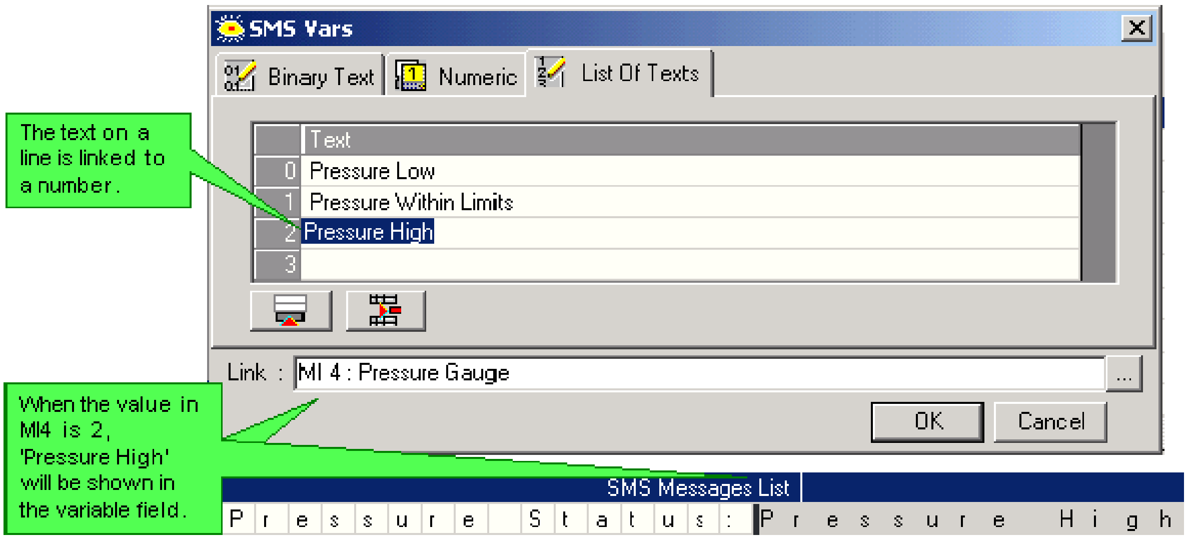
A Number Variable enables you to:

- Show any numeric value within a message.
- Control the format in which that value is shown, including the placement of a decimal point and leading zeros.
- Use Linearization to show a converted value, such as an analog temperature converted to degrees Celsius.


Prevents the entry of an 'illegal' value.
Set the range for a 'legal' value.
Use these to determine:
▪ the number of digits to be displayed.
▪ decimal point position.
The numeric value will be shown in the variable field.
Allows you to convert values, for example from analog to digital.
This is the range for the 'y' axis value.
This is the range for the 'x' axis value.
SMS Vars
Binary Text
Numeric
Entry Limits
Min 39
Max 42
Format
3 . 2
999.99
Leading zeros
Linearization
Display
Max 0
Min 100
Op Value
Min 0
Max 1023
Link : MI 10 : Requested Temperature
OK
Cancel
SMS Messages List
Request Temperature 999.99

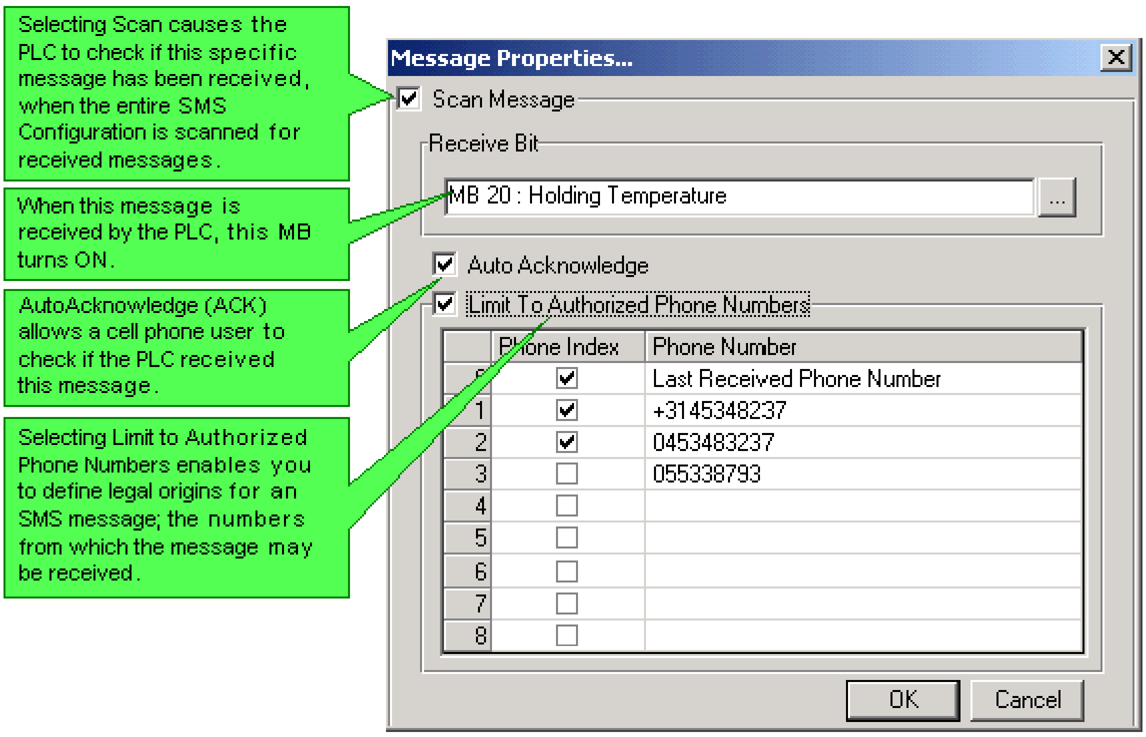
List of Texts: by Pointer

This type of variable contains numbered lines of text. You link the Variable to an operand. The value within that operand 'points' to the number of a line within the list. When the operand value is equal to a particular line number, the text of that line is shown in the Display.


The text on a line is linked to a number.
When the value in MI4 is 2, 'Pressure High' will be shown in the variable field.
SMS Vars
Binary Text
Numeric
List Of Texts
Text
0 Pressure Low
1 Pressure Within Limits
2 Pressure High
3
Link : MI 4 : Pressure Gauge
OK
Cancel
SMS Messages List
Pressure Status: Pressure High

Message Properties

This defines how the PLC deals with a specific SMS message that is **received** by the system. Note that a SMS Scan FB must be placed in the Ladder to enable the PLC to check a specific SMS Configuration for received messages.



Selecting Scan causes the PLC to check if this specific message has been received, when the entire SMS Configuration is scanned for received messages.

When this message is received by the PLC, this MB turns ON.

AutoAcknowledge (ACK) allows a cell phone user to check if the PLC received this message.

Selecting Limit to Authorized Phone Numbers enables you to define legal origins for an SMS message; the numbers from which the message may be received.

Message Properties...

☑ Scan Message

Receive Bit: MB 20 : Holding Temperature

☑ Auto Acknowledge

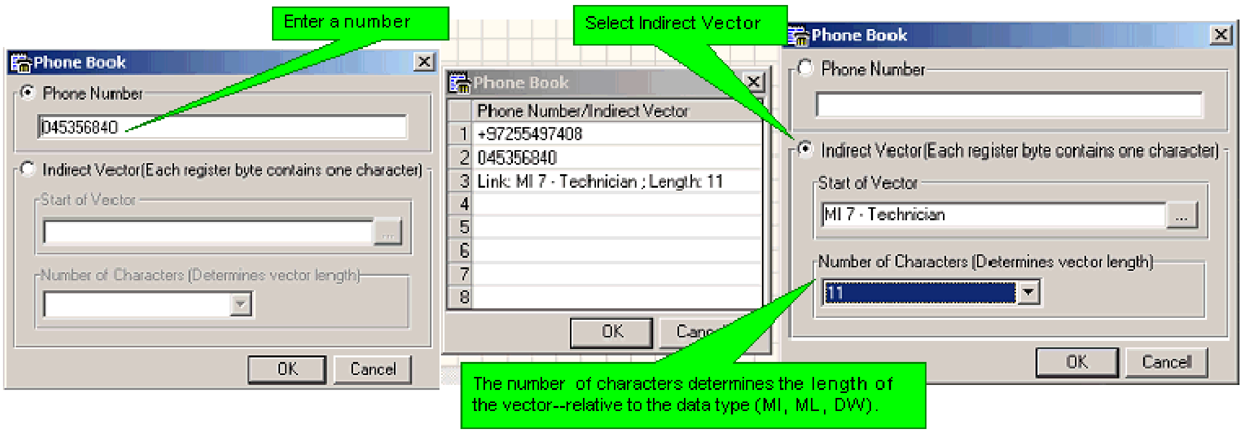
☑ Limit To Authorized Phone Numbers

	Phone Index	Phone Number
0	☑	Last Received Phone Number
1	☑	+3145348237
2	☑	0453483237
3	☐	055338793
4	☐	
5	☐	
6	☐	
7	☐	
8	☐	

Phone Book

Click on a line to either:

- Directly enter a phone number in the phone book, or
- Use a vector of registers to indirectly dial a number. HMI Keypad Entry Variables can then be linked to this vector, enabling a user to dial a number from the controller keypad via HMI Entry Variables.



Enter a number

Phone Book

(●) Phone Number: 045356840

( ) Indirect Vector(Each register byte contains one character)

Start of Vector:

Number of Characters (Determines vector length):

Phone Book

	Phone Number/Indirect Vector
1	+97255497408
2	045356840
3	Link: MI 7 - Technician ; Length: 11
4	
5	
6	
7	
8	

Select Indirect Vector

Phone Book

( ) Phone Number:

(●) Indirect Vector(Each register byte contains one character)

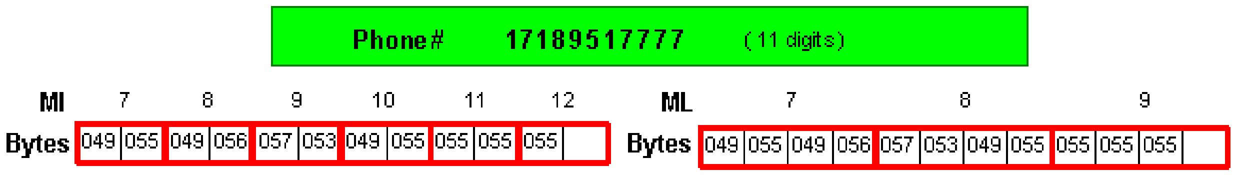
Start of Vector: MI 7 - Technician

Number of Characters (Determines vector length): 11

The number of characters determines the length of the vector--relative to the data type (MI, ML, DW).

Indirect Vector

Each register byte contains 1 character. Note that the byte actually contains the **ASCII** value of the desired numeral. For example, the ASCII value of 1 is 049.



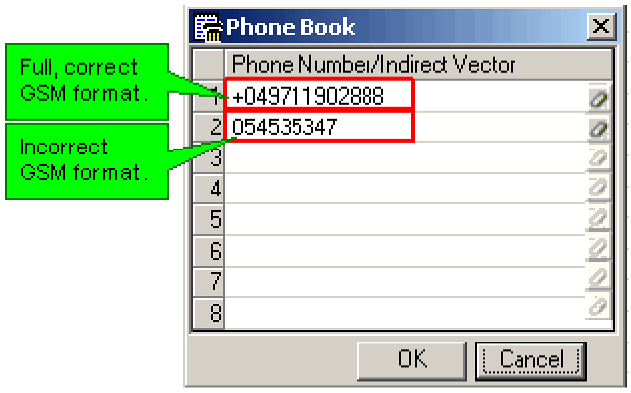
Phone# 17189517777 (11 digits)

MI	7		8		9		10		11		12	
Bytes	049	055	049	056	057	053	049	055	055	055	055	

ML	7				8				9			
Bytes	049	055	049	056	057	053	049	055	055	055	055	

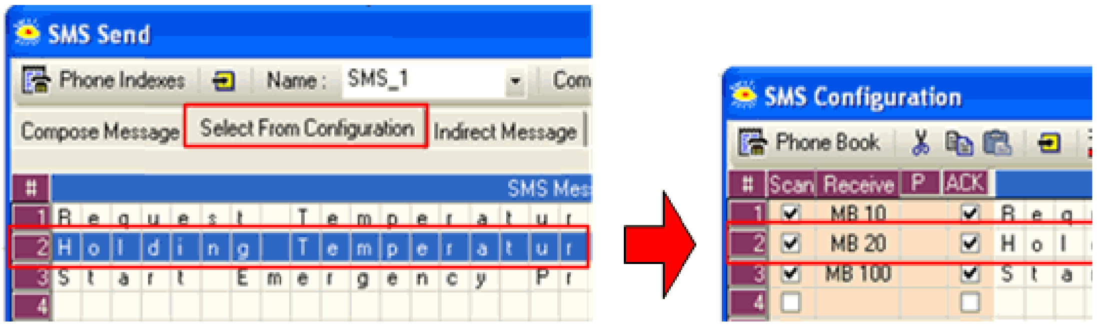
International Cell Phone Number Format

To work with international phone numbers, use the full GSM format, including the '+' in front of the country code.

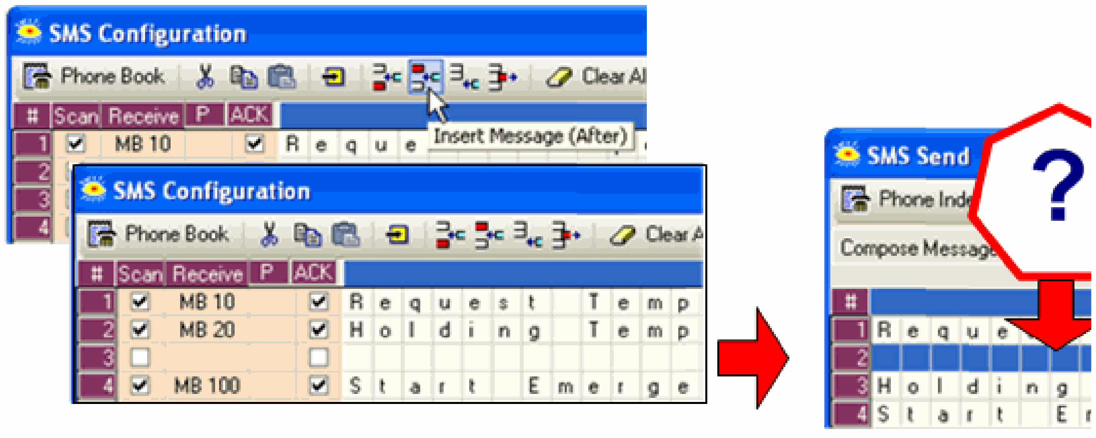


	Phone Number/Indirect Vector
1	+049711902888
2	054535347
3	
4	
5	
6	
7	
8	

Message Index Numbers
The messages in the Configuration are numbered consecutively. Inserting and deleting messages causes the index numbers to increment and decrement.
If your application contains Send functions that send messages from the Configuration, you must edit the message index number accordingly.
Example
In the following figure, the Send function is set to send the message with Index # 2.



In the next figure, the user inserted a line, causing the index number to increment. However, the user did not edit the Send function, and message Index # is now blank.

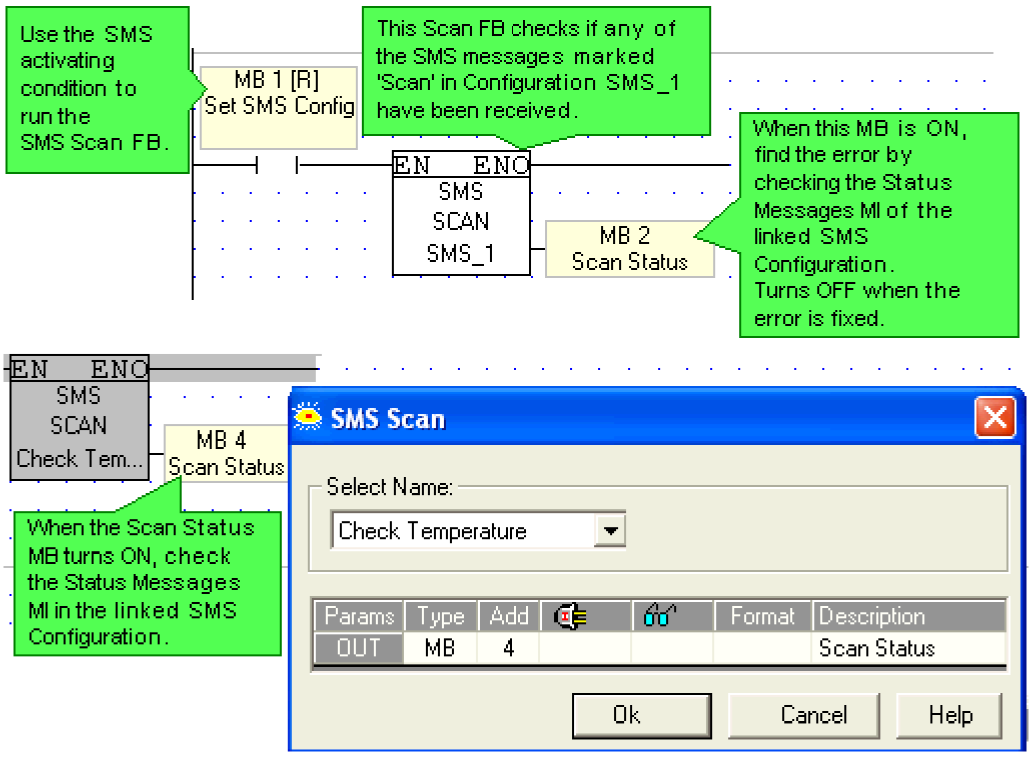


SMS: Scan

To enable the controller to receive messages, place an SMS Scan FB in your application and link it to a Configuration. When activated, this causes the controller to scan the GSM modem's SIM card for incoming SMS messages.

Before you can receive an SMS, you must initialize a COM port to use a GSM modem, create an SMS Configuration, and set conditions as explained in Using SMS Messaging.

Note •	Communications cannot flow through the port during initialization. To avoid conflicts in your program, use the Modem Initialization Status SBs.
•	Although the PLC can send Binary Text, Numeric, and List of Texts variables, it can only receive **Numeric variables**.
•	Use an MB activated by the SMS Configuration FB to activate the Scan, as shown in Using SMS Messaging


Use the SMS activating condition to run the SMS Scan FB.

This Scan FB checks if any of the SMS messages marked 'Scan' in Configuration SMS_1 have been received.

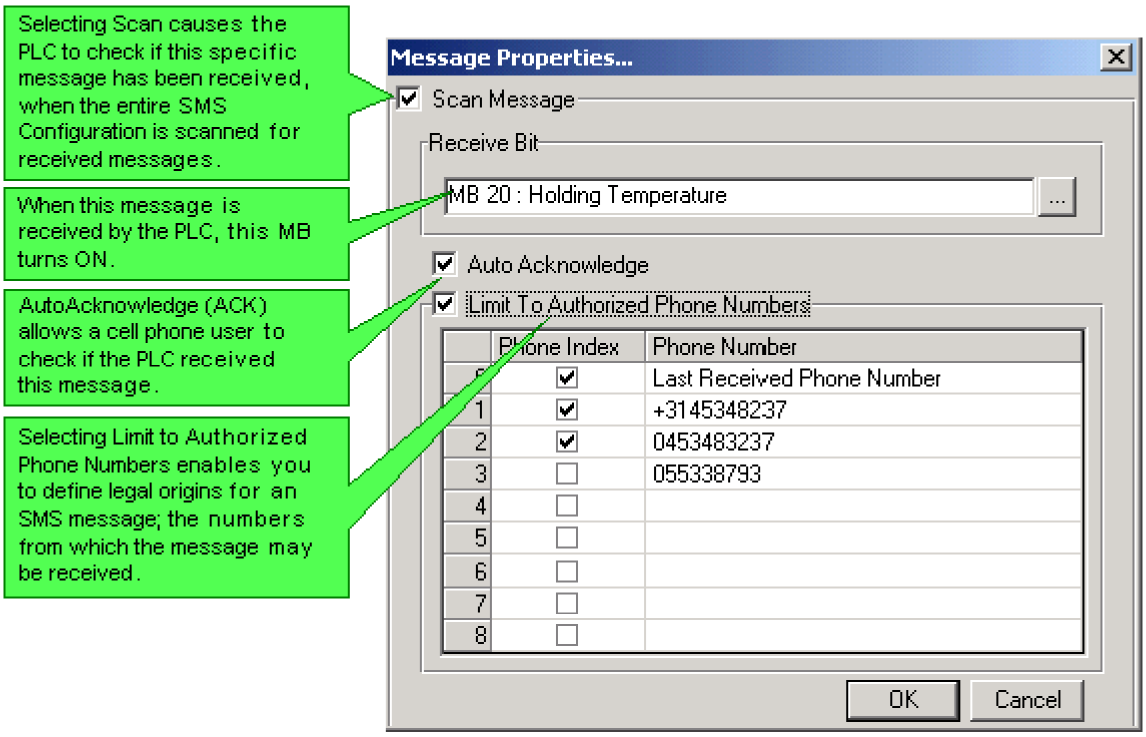
When this MB is ON, find the error by checking the Status Messages MI of the linked SMS Configuration. Turns OFF when the error is fixed.

When the Scan Status MB turns ON, check the Status Messages MI in the linked SMS Configuration.

Select Name: Check Temperature

Params	Type	Add			Format	Description
OUT	MB	4				Scan Status

When the PLC receives a message, the Message Properties defined in the SMS configuration for that message define how it will be processed.


Selecting Scan causes the PLC to check if this specific message has been received, when the entire SMS Configuration is scanned for received messages.

When this message is received by the PLC, this MB turns ON.

AutoAcknowledge (ACK) allows a cell phone user to check if the PLC received this message.

Selecting Limit to Authorized Phone Numbers enables you to define legal origins for an SMS message; the numbers from which the message may be received.

Message Properties...

☑ Scan Message

Receive Bit: MB 20 : Holding Temperature

☑ Auto Acknowledge

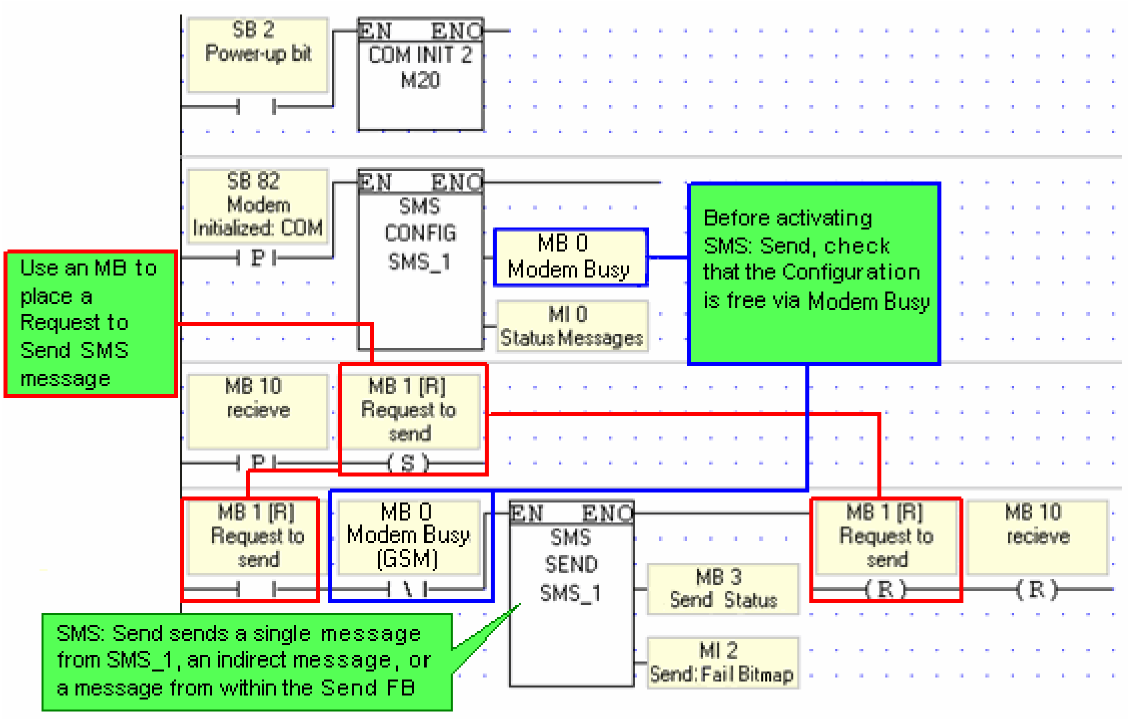
☑ Limit To Authorized Phone Numbers

	Phone Index	Phone Number
0	☑	Last Received Phone Number
1	☑	+3145348237
2	☑	0453483237
3	☐	055338793
4	☐	
5	☐	
6	☐	
7	☐	
8	☐	

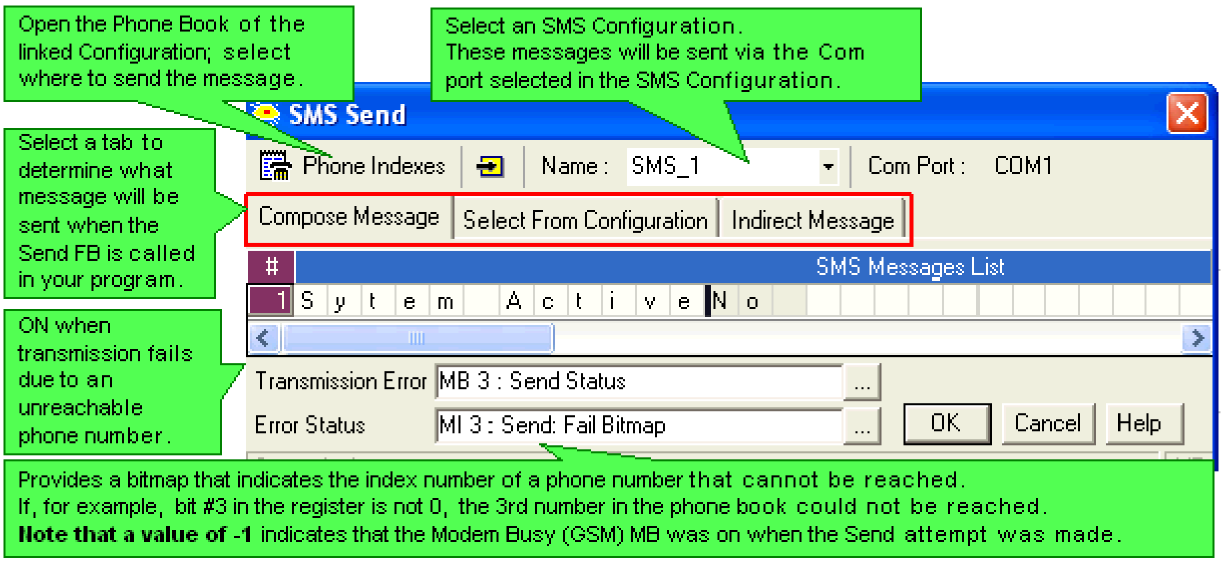
SMS: Send

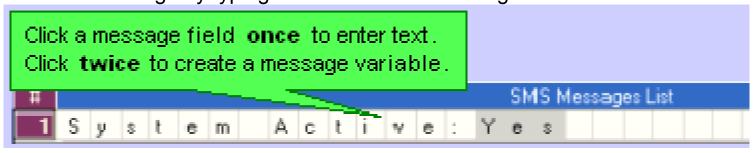
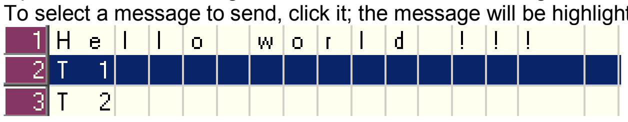
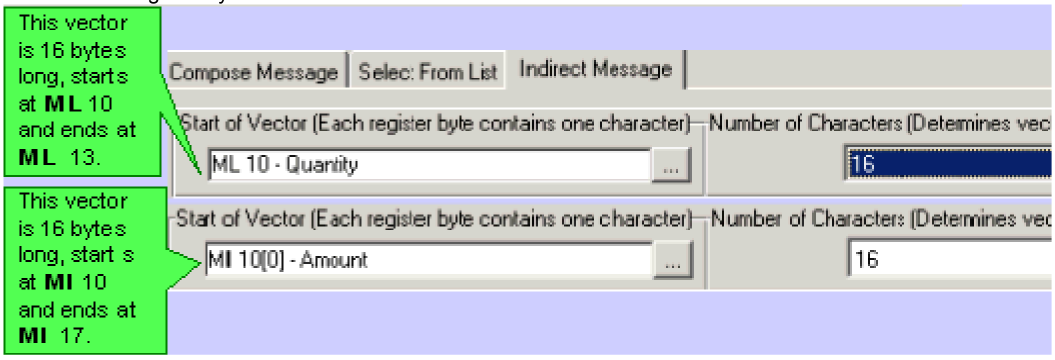
To send an SMS, place an SMS Send operation in your Ladder application. Before you can send an SMS, you must initialize a COM port to use a GSM modem, create an SMS Configuration, and set conditions as explained in Using SMS Messaging.

Note •	Communications cannot flow through the port during initialization. To avoid conflicts in your program, use the Modem Initialization Status SBs.
•	Before using a modem, prepare it as described in the help topic PLC-side Modems.
•	If the SMS Configuration is busy, messages cannot be received. Avoid conflicts by using a negative contact, linked to the Configuration's Modem is Busy MB, as an activating condition for SMS: Send.



Note •	You can send only one SMS message at a time, but you may send it to multiple phone numbers.
•	The Send Status MB turns on if there are more than one Send operations, and the PLC has not finished processing the first Send before the second Send is activated. In this case, Modem Busy MB of the SMS: Configuration bit will be ON, causing the Send Fail Bitmap to contain a value of -1.


Open the Phone Book of the linked Configuration; select where to send the message.
Select an SMS Configuration. These messages will be sent via the Com port selected in the SMS Configuration.
Select a tab to determine what message will be sent when the Send FB is called in your program.
ON when transmission fails due to an unreachable phone number.
Provides a bitmap that indicates the index number of a phone number that cannot be reached. If, for example, bit #3 in the register is not 0, the 3rd number in the phone book could not be reached. Note that a value of -1 indicates that the Modem Busy (GSM) MB was on when the Send attempt was made.
SMS Send
Phone Indexes
Name : SMS_1
Com Port : COM1
Compose Message
Select From Configuration
Indirect Message
SMS Messages List
Transmission Error MB 3 : Send Status
Error Status MI 3 : Send: Fail Bitmap
OK
Cancel
Help

Compose Message	Opens a blank message field. Create a message by typing characters and attaching variables.  Click a message field once to enter text. Click twice to create a message variable. SMS Messages List
Select from Configuration	Opens the list of messages contained in the linked Configuration. To select a message to send, click it; the message will be highlighted. 
Indirect Message	Enables a vector of register values to be sent as ASCII characters. **Note** • Each register byte contains one character.  This vector is 16 bytes long, starts at ML 10 and ends at ML 13. This vector is 16 bytes long, start s at MI 10 and ends at MI 17. Compose Message Selec From List Indirect Message Start of Vector (Each register byte contains one character) Number of Characters (Detemines vec ML 10 - Quantity 16 MI 10[0] - Amount 16

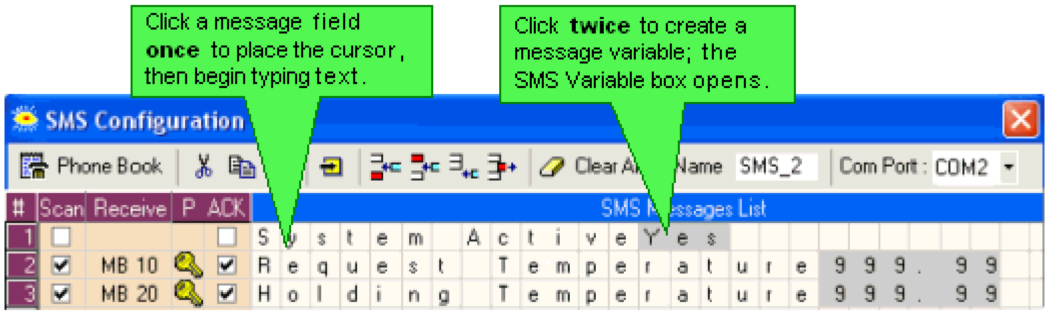
Note •	When you create messages in a Configuration, VisiLogic does not allow you to begin a message with a fixed text character that is not a numeral (0-9) or a number sign (#). These limitations do not apply to messages that you compose in an SMS Send FB, or to Indirect Messages. These may begin with variables or with any alphanumeric character.
•	A message containing only English characters may contain up to 160 characters. A message containing non-English characters may contain up to 70

	characters.
•	Phone Indexes is based on the Phone Book in the linked SMS configuration. Index numbers are 0 to 8, note that 0 is the index of the last number called.
•	SB 184 SMS: Transmission Succeeded and 185 SMS: Transmission Failed are both reset when any message is sent to any phone number. After the message has been processed, the Relevant SB will turn ON.
•	You can send messages in ASCII text format by turning ON SB 279: ASCII Text message format,
•	If forced display messages are supported by the cell phone receiving the SMS, you can send a forced display message by turning ON SB 280: Force Message Display on Cell Phone.

SMS Variables

Variables can be included in Configuration or Send FB messages. A single SMS message can contain both fixed text and up to 10 variables. Note that SMS variables are not related to HMI variables.

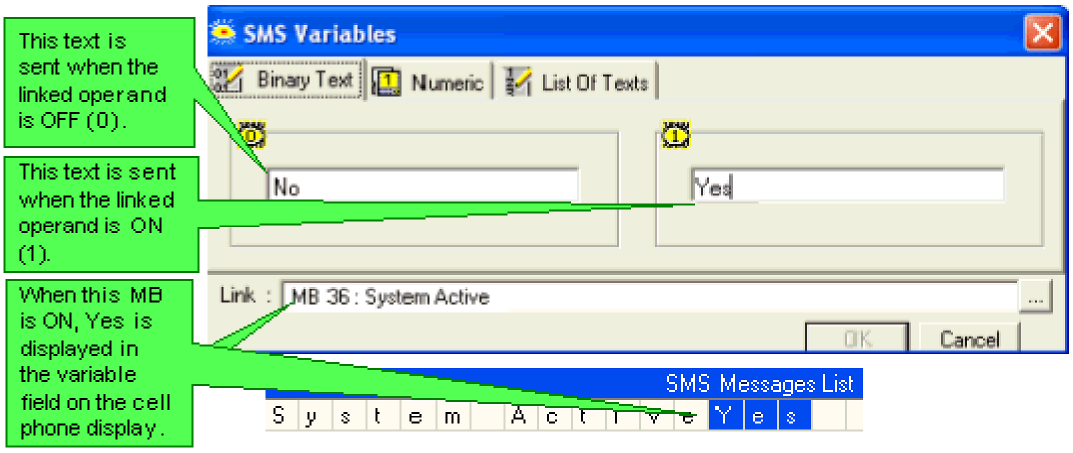
Although the PLC can send Binary Text, Numeric, and List of Texts variables, it can only receive **Numeric** variables.


Click a message field **once** to place the cursor, then begin typing text.
Click **twice** to create a message variable; the SMS Variable box opens.
SMS Configuration
Phone Book
Clear All
Name SMS_2
Com Port : COM2
SMS Messages List

#	Scan	Receive	P	ACK	SMS Messages List
1	☐			☐	System Active Yes
2	✓	MB 10		✓	Request Temperature 999.99
3	✓	MB 20		✓	Holding Temperature 999.99

Binary Variable

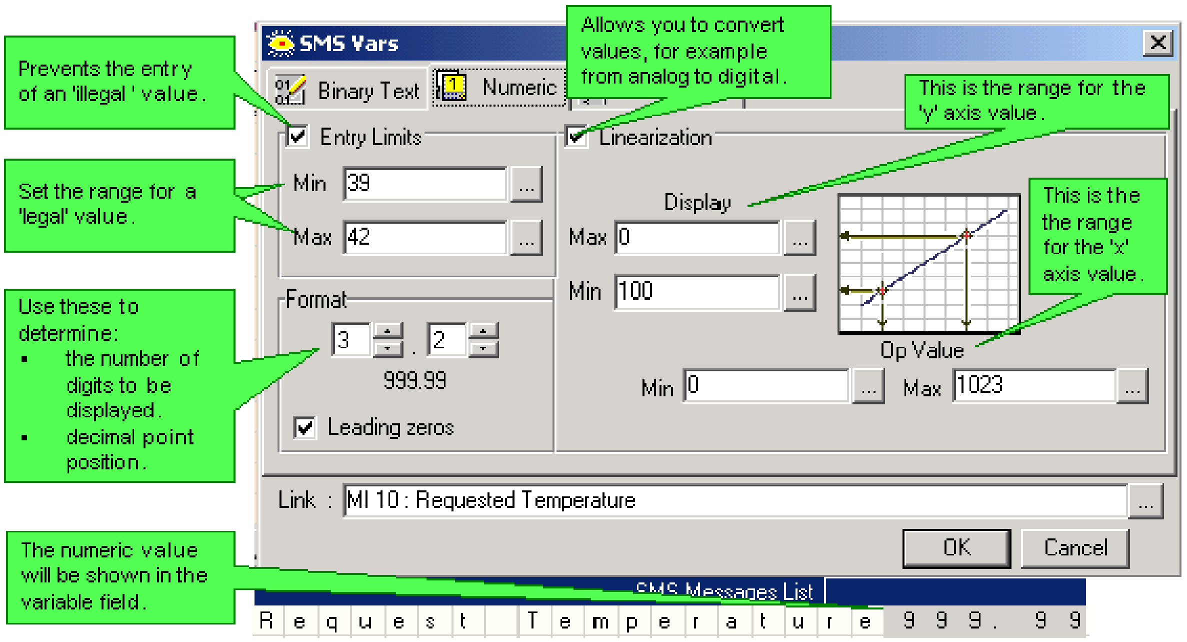
This type of variable displays different text in the SMS variable field according to the status of a bit operand. The value currently visible is the last value sent.


This text is sent when the linked operand is OFF (0).
This text is sent when the linked operand is ON (1).
When this MB is ON, Yes is displayed in the variable field on the cell phone display.
SMS Variables
Binary Text
Numeric
List Of Texts
No
Yes
Link : MB 36 : System Active
OK
Cancel
SMS Messages List
System Active Yes

Numeric Variable

A Number Variable enables you to:

- Show any numeric value within a message.
- Control the format in which that value is shown, including the placement of a decimal point and leading zeros.
- Use Linearization to show a converted value, such as an analog temperature converted to degrees Celsius.

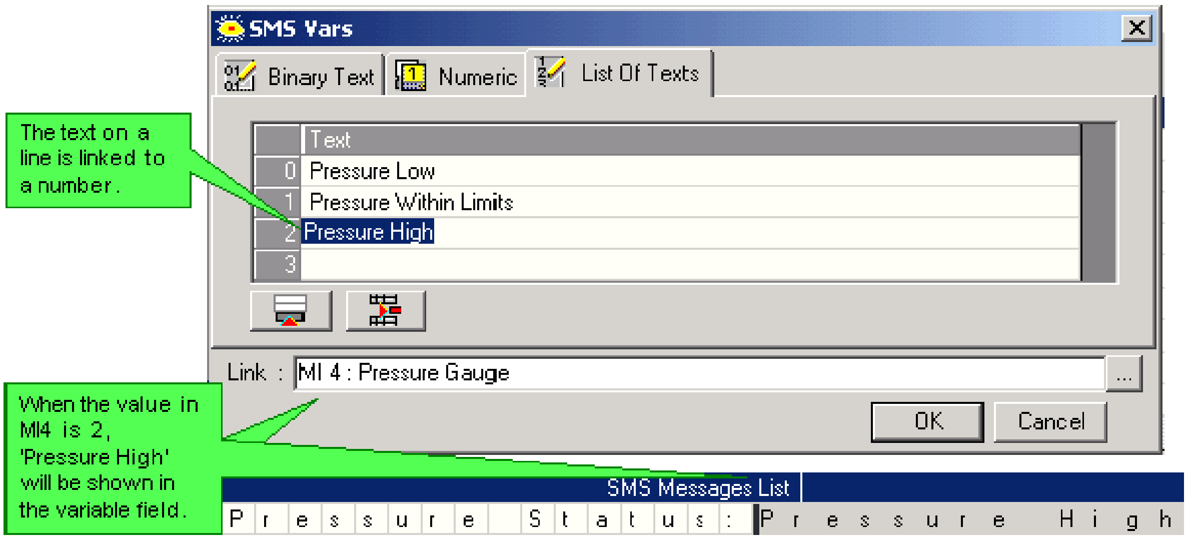

Prevents the entry of an 'illegal' value.
Set the range for a 'legal' value.
Use these to determine:
▪ the number of digits to be displayed.
▪ decimal point position.
The numeric value will be shown in the variable field.
Allows you to convert values, for example from analog to digital.
This is the range for the 'y' axis value.
This is the range for the 'x' axis value.
SMS Vars
Binary Text
Numeric
Display
Op Value
OK
Cancel
SMS Messages List

Entry Limits ✓
Min 39
Max 42
Format 3 . 2
999.99
Leading zeros ✓
Linearization ✓
Display Max 0
Display Min 100
Op Value Min 0 Max 1023
Link : MI 10 : Requested Temperature

Request Temperature 999.99

List of Texts: by Pointer

This type of variable contains numbered lines of text. You link the Variable to an operand. The value within that operand 'points' to the number of a line within the list. When the operand value is equal to a particular line number, the text of that line is shown in the Display.


The text on a line is linked to a number.
When the value in MI4 is 2, 'Pressure High' will be shown in the variable field.
SMS Vars
Binary Text
Numeric
List Of Texts
OK
Cancel
SMS Messages List

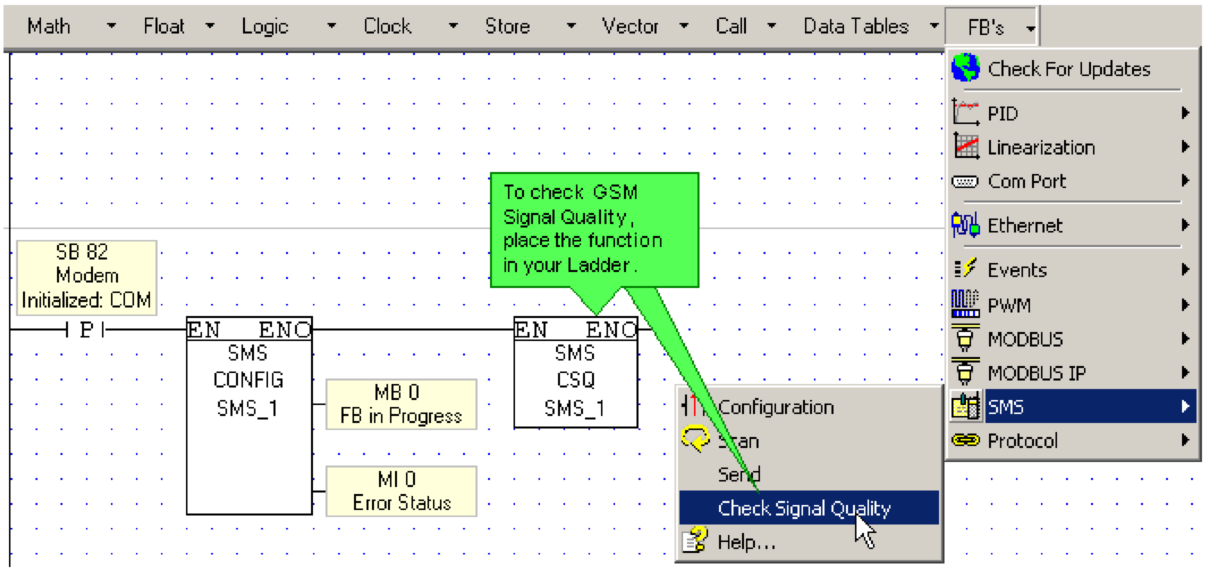
	Text
0	Pressure Low
1	Pressure Within Limits
2	Pressure High
3	

Link : MI 4 : Pressure Gauge

Pressure Status: Pressure High

SMS: Check GSM Signal Quality

The signal quality may be checked at any point after SMS Configuration.



The signal quality value is placed in **SI 185 GSM Signal Quality**.

A value of -1(FFFF)signifies a modem error. This may be due to a weak signal; try repositioning the antenna. If this has no effect, check the modem.

Send SMS messages from a GSM cell phone

To send SMS messages from your cell phone to your PLC, you must:

- Create and download a project to your PLC that includes an SMS Configuration, set Message Properties, and define conditions as described in the topic Using SMS Messaging.
- Write an SMS message in your cell phone.
- Send the message to the PLC's GSM modem

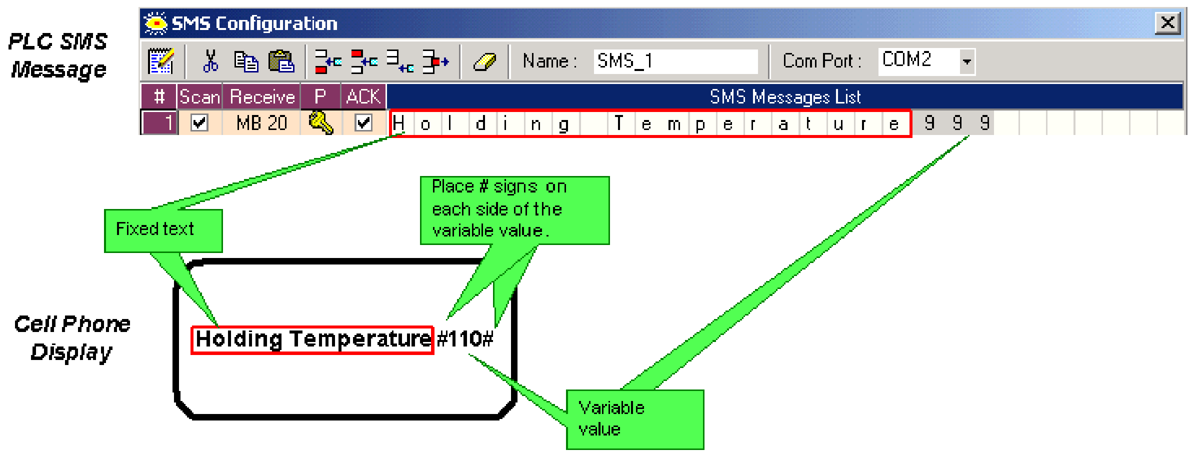
Note •	You can only send messages that are already part of an SMS Configuration in the PLC.
•	If the Limit to Authorized Phone Numbers option is selected in the SMS configuration, the cell phone number must be in the list.

Writing SMS messages in your cell phone

You write an SMS message using your cell phone keypad. Make sure that:

- The fixed text in your cell phone is identical to the message in the PLC's SMS Configuration **in every detail**: spaces, characters--and note that characters are case-sensitive.
- You bracket variable values with number signs (#) as shown below. These signs '#' do **not** count as spaces.
- The variable field in the SMS message is big enough to hold the value.

The figure below shows the same SMS message: as it appears on a cell phone display, and as it appears in the PLC's SMS Configuration.



When you send this message from your cell phone, the value 110 will be written into the variable in the PLC.

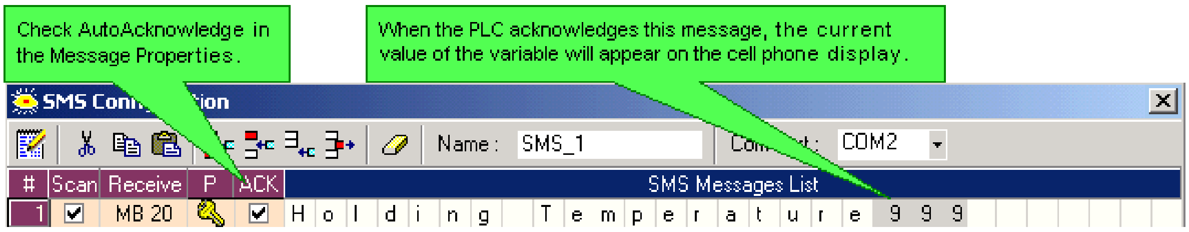
Sending the message to the PLC

1. Enter the number of the PLC's GSM modem exactly as you would enter any GSM cell phone number, then send the message.

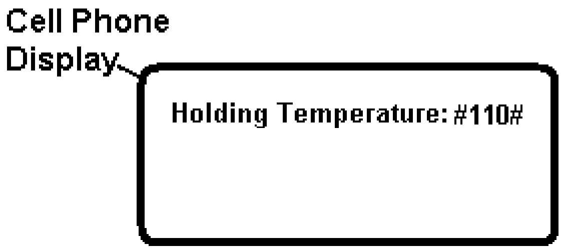
Checking that the PLC has received the SMS message

You can check if the PLC received your message by using the Acknowledge feature:

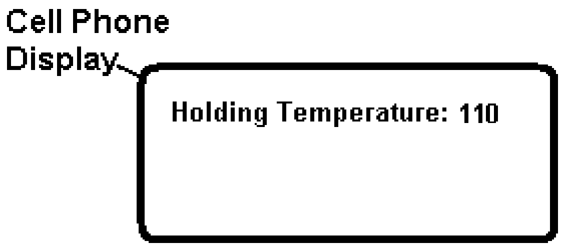
1. Select 'Acknowledge' in Message Properties, the **ACK** box is checked as shown below.


Check AutoAcknowledge in the Message Properties.
When the PLC acknowledges this message, the current value of the variable will appear on the cell phone display.
SMS Con ion
Name: SMS_1
Com t: COM2
# Scan Receive P ACK SMS Messages List
1 ✓ MB 20 ✓ H o l d i n g T e m p e r a t u r e 9 9 9

2. Use your cell phone to send the message "**Holding Temperature:#110#"** to the PLC.



3. The PLC receives this SMS message; AutoAcknowledge causes the PLC to immediately return the message to your cell phone, together with the current variable value.
4. You can now view this SMS message on your cell phone display, together with changes in the variable value.



Note that although the PLC can send SMS messages that include Numeric, Binary, and List variables, the PLC can only receive Numeric variables.

SMS Operands

Operands linked to SMS FBs should be assigned Power-up Values; bits should be reset, and registers initialized to 0.

SMS: Configuration FB

Modem Busy (GSM) Reset when **SMS: Configuration** is activated.	**MB**	**Turns ON when:** • **SMS: Send** FB is activated. • **Check GSM signal** FB is active • A **Scan** FB is processing incoming messages. • An SMS is being Auto-acknowledged. • The PLC is communicating data with another application, such as VisiLogic.	**Turns OFF when** • **Send** process is complete. • When messages received from a **Scan** FB have been processed.
Status Messages Initialized to 0 when **SMS: Configuration** is activated.	**MI**	• Updated at the end of all operations using the **SMS: Configuration**; **Send**, **Scan**, and **Check GSM Signal** FBs. • Indicates error status for **Send**, **Scan**, and **Check GSM Signal** FBs. • Current value always shows the most **<u>recent</u>** error status. Value Message 0 No error 1 Message received from a phone number that is not in the phone book, or the number is not in the correct format 2 (Send only) Non-existent SMS message index number 3 SMS received from unauthorized phone number 4 (Scan only)The SMS message received does not exist in the SMS configuration 5 Modem TimeOut time exceeded: no reply 6 (Scan only) Received Variable Mismatch. Variable: - Does not exist in the SMS configuration, or - Is not in the correct format, or - Exceeds the range set for the variable 7 Modem Reply Error 8 Unknown Modem Reply 9 (Send only)Either the phone number or the SMS message is in the incorrect format and may not be transmitted	

SMS Scan FB

Scan Status Initialized to 0 the first time the **SMS Scan** is called	**MB**	**Turns ON when:** • An error occurs during the current **Scan**. The Error message is indicated in the **Status Messages** MI of the **SMS: Configuration.**	**Turns OFF when** • A **Scan** operation is successfully completed.

SMS: Send FB

Send Status	**MB**	**Turns ON when:** • The **Send** operation has been completed, and one or more of the SMS messages has **<u>not</u>** been successfully sent. • This MB turns on if there are more than one Send operations, and the PLC has not finished processing the first Send before the second Send is activated. In this case, **Modem Busy (GSM)** MB of the **SMS: Configuration** will be ON, causing the **Send Fail Bitmap** will contain a value of -1. -The Error message is indicated in the **Error Status MI** of the **SMS: Configuration** linked to the **Scan** FB -The **Send: Fail Bitmap** indicates which phone numbers have not received the SMS.	**Turns OFF when** • All messages in the **SMS: Send** have been sent to all of the phone numbers.

Send: Fail Bitmap	**MI**	• Provides a bitmap that indicates the index number of a phone number that cannot be reached. If, for example, bit #3 in the register is not 0, the 3rd number in the phone book could not be reached. • The bits are initialized to 0 when the Send FB is activated. As calls are made, each **failure** causes the relevant bit to turn ON. • Note that: - Phone Indexes is based on the Phone Book in the linked SMS configuration. Index numbers are 0 to 8, note that 0 is the index of the last number called. - a value of -1 indicates that the **Modem Busy** MB of the **SMS: Configuration** was ON when the Send attempt was made.

SMS System Operands

The status of these SBs changes to reflect the status of each SMS message sent by the system, including when messages are auto-acknowledged.

When the Send process begins, for each and every message, both SB 184 and 185 are OFF. After the message is sent, the relevant bit turns ON, indicating the success or failure of that message.

Each port has a Succeed and Fail SB. When the Send process begins from a particular COM port, for each and every message, both the Succeed and Fail SB turn OFF.
If the message is sent successfully, the bit turns ON, indicating the success or failure of that message.
If the message fails, when TimeOut is exceeded or because the modem reports an error, the bit remains OFF.
Operands that are linked by the user to SMS FBs may be found in the topic SMS Operands.

SB	Description	Turns ON when:	Turns OFF when:
184	SMS: Transmission Succeeded, COM1 (ACK)	Transmission succeeds	Transmission begins
185	SMS: Transmission Failed, COM1 (NACK)	Transmission fails	
186	SMS: Transmission Succeeded, COM2 (ACK)	Transmission succeeds	
187	SMS: Transmission Failed, COM2 (NACK)	Transmission fails	
188	SMS: Transmission Succeeded, COM3 (ACK)	Transmission succeeds	
189	SMS: Transmission Failed, COM3 (NACK)	Transmission fails	

Use these together with SI 198 and 199 to write incoming SMS messages to a vector of operands. This does not affect the function of the SMS message function blocks.

SB	Description	Turns ON when:	Turns OFF when:
198	Record the Received SMS Message Length	If SB 199 is set, , SB 198 is set when a message is received	User Program

199	Save SMS to Memory Vector	User Program

Use these together with SB 198 and 199 to write incoming SMS messages to a vector of operands. This does not affect the function of the SMS message function blocks.

SI	Description	Value
198	Received SMS Message Length	Shows the length of the message in bytes The data remains until the vector is overwritten
199	SMS to Memory Vector - start of vector	The SMS message data is written starting from this address. To write to a vector of XIs, enter a negative value

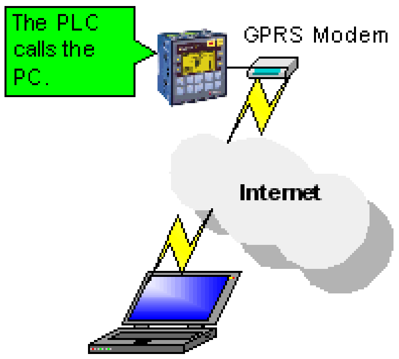
GPRS

GPRS Overview

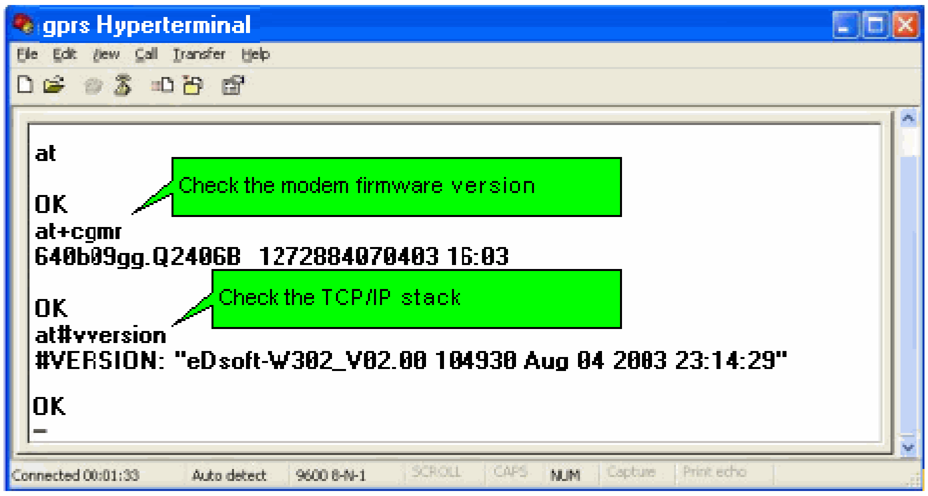
GPRS is a wireless data transmission service offered by some cellular providers. When a Unitronics' PLC is connected to a GPRS modem, the PLC can establish a data connection with a remote PC connected to the Internet and transmit IP packets of data over the GPRS cellular network.

You can then use your PC to access the PLC via VisiLogic, Remote Access or other communication .dll, as well as log PLC data via the DataXport utility. You can also send e-mail via GPRS modem.

The figure below shows GPRS communication elements.



Notes •	The PLC must be connected to a GPRS modem. GPRS service must be supplied by the user's cellular service provider.
•	To enable any GPRS operations to run, the PLC application must include GPRS Configuration and GPRS Run FBs.
•	Y ou must prepare both the PLC and PC side modems by carrying out the Prepare PLC-side Modem procedure detailed in the topic PLC-side modems, and the Initialize PC Modem detailed in the topic PC-side modems.
•	To send e-mail via GRPS modem,when you prepare the modem via Modem Services, set it to a baud rate of 9600. In additon, the COM Init function should be set to 9600.
•	Note that regarding Enhanced Vision models, as of OS 3.1.19 GPRS is no longer supported for Wavecom and Sony Ericsson modems. To view the list of supported modems, open the COM Init function and select GSM modem as shown in the section below, Using GPRS.
•	Unitronics has tested the following GPRS modems with Unitronic's PLCs: Wavec om Fastrack WM15467 (GPRS+TCP/IP 900/1800 MHz), Firmware version: 640b09gg.Q2406B 1272884070403, TCP/IP stack: eDsoft-W302_V02.00 104930 You can use Hyperterminal to: Check the modem's firmware version, via command AT+CGMR. Check the TCP/IP Stack, via command AT#VVersion.


gprs Hyperterminal
File Edit View Call Transfer Help
Check the modem firmware version
Check the TCP/IP stack
Connected 00:01:33 Auto detect 9600 8-N-1 SCROLL CAPS NUM Capture Print echo

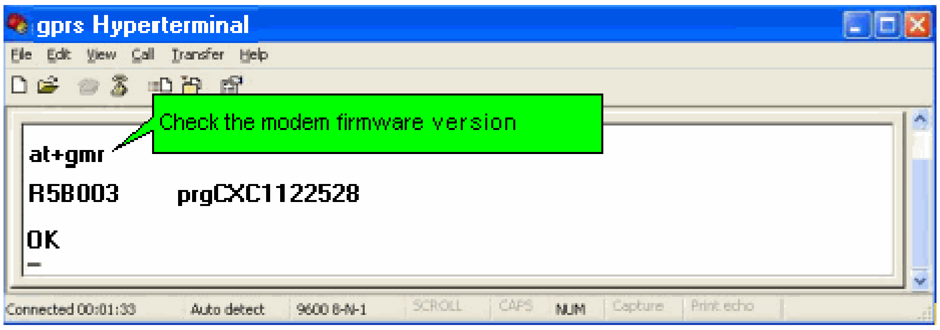
```
at

OK
at+cgmr
640b09gg.Q2406B 1272884070403 16:03

OK
at#vversion
#VERSION: "eDsoft-W302_V02.00 104930 Aug 04 2003 23:14:29"

OK
```

Sony Ericsson GT47
Firmware version: R5B003 prgCXCC1122528
You can use Hyperterminal to:
Check the modem's firmware version, via command AT+GMR.


gprs Hyperterminal
File Edit View Call Transfer Help
Check the modem firmware version
Connected 00:01:33 Auto detect 9600 8-N-1 SCROLL CAPS NUM Capture Print echo

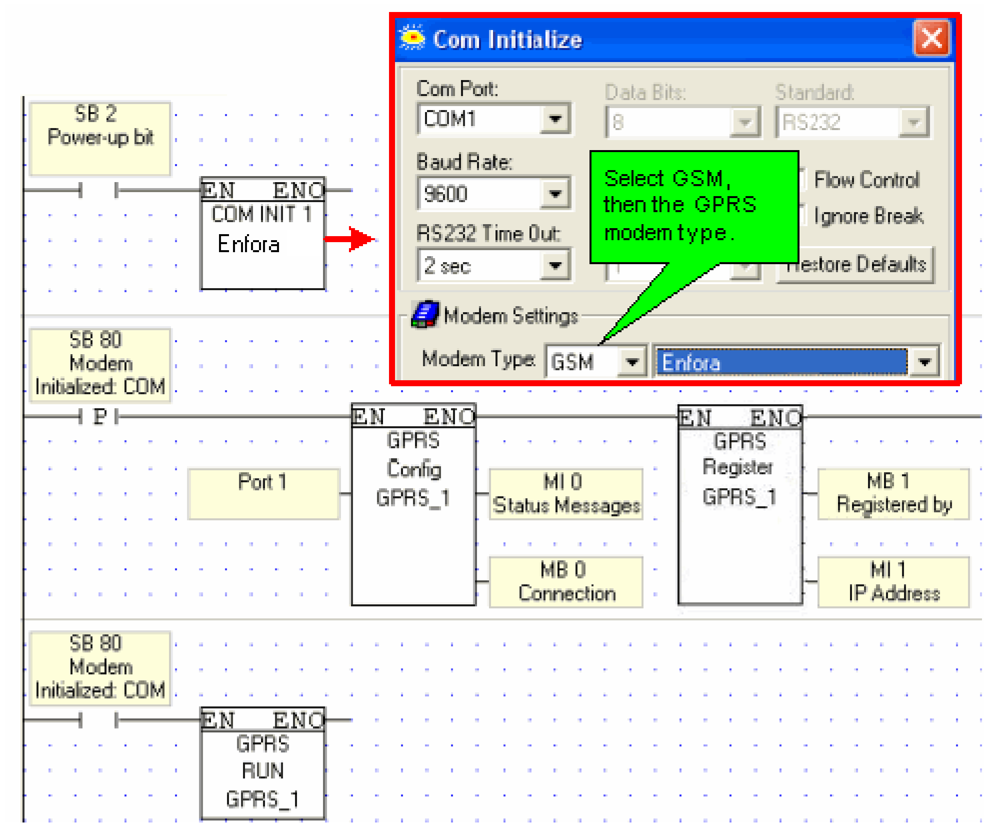
```
at+gmr
R5B003      prgCXC1122528

OK
```

•	The PLC must initiate the GPRS data link by calling the PC. A sample application showing how to establish PC-PLC communications via GPRS is available for currently supported modems. This application is intended to be used as a template, for you to simply copy and adapt to your own requirements. Sample applications are located under Help>Examples.
•	Wavecom modems do not support the End Call function. In order to end a call, use the Unregister from Network function, and then re-register if required.
	The Sony Ericsson GT47 GPRS modem must be used in conjunction with an appropriate cable and 5 pin connector.
•	Not supported by the V120-12 series.

Using GPRS

The PLC's Ladder application must contain the conditions and elements shown below.



Com Port: COM1
Data Bits: 8
Standard: RS232
Baud Rate: 9600
RS232 Time Out: 2 sec
Modem Type: GSM Enfora

FB Name	Purpose
COM Port Init FB	Configure this FB to initialize the PLC COM Port connected to the GPRS-modem. Within the COM Init FB, select the GSM modem type, and then the actual GPRS modem type, such as the Sony Ericsson GT47. COM Port Init is usually a Power-up task. **Note** • To learn how to prepare the modem for PLC use, check the topic PLC-side Modems.
GPRS Configuration	The Configuration is linked to the serial port initialized via the COM Port Init FB. **Note** • The activating conditions should include the appropriate Modem Initialized SB for the COM Port.
GPRS Register to Network	Register must follow the GPRS Configuration. Connect provides the parameters that enable the PLC to connect to the Internet via the cellular network.
GPRS Run	This element enables data communications via GPRS. **Note** • The activating conditions should include the appropriate Modem Initialized SB for the COM Port.

Downloading OS via GPRS

Enhanced Vision controllers using Boot 2.00 and higher support OS download via GPRS modem.

Notes •	The TC65 does not support OS download.
	The modem must be connected to COM2.

Once the download begins, the controller enters Boot mode. Any interruption in communication may result in the controller being 'stuck' in boot mode, without an installed OS. For this reason, **it is it is recommended that**

someone be next to the PLC during the OS download in order to attend to any potential problems.

FB Operations

GPRS Operations are located on the FBs menu.

Configuration

Register to Network

Run

Start/End Call

Listen/Stop Listening Remote Device

Unregister (Disconnect) from Network

Check Signal Quality

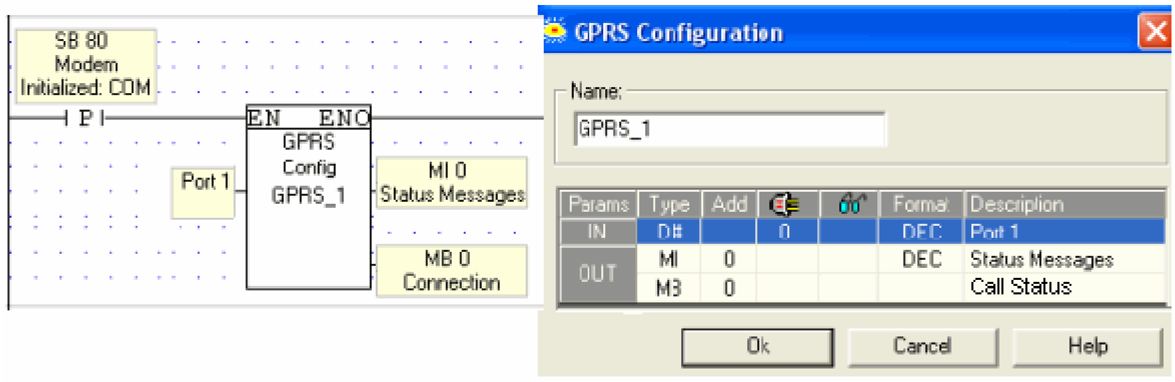
GPRS Operands & Status Messages

GPRS Configuration

The GPRS Configuration must be included in all GPRS Ladder applications.

The Configuration is linked to the serial port connected to the GPRS modem and initialized via the COM Port Init FB.

Note • The activating conditions should include the appropriate Modem Initialized SB for the COM Port.



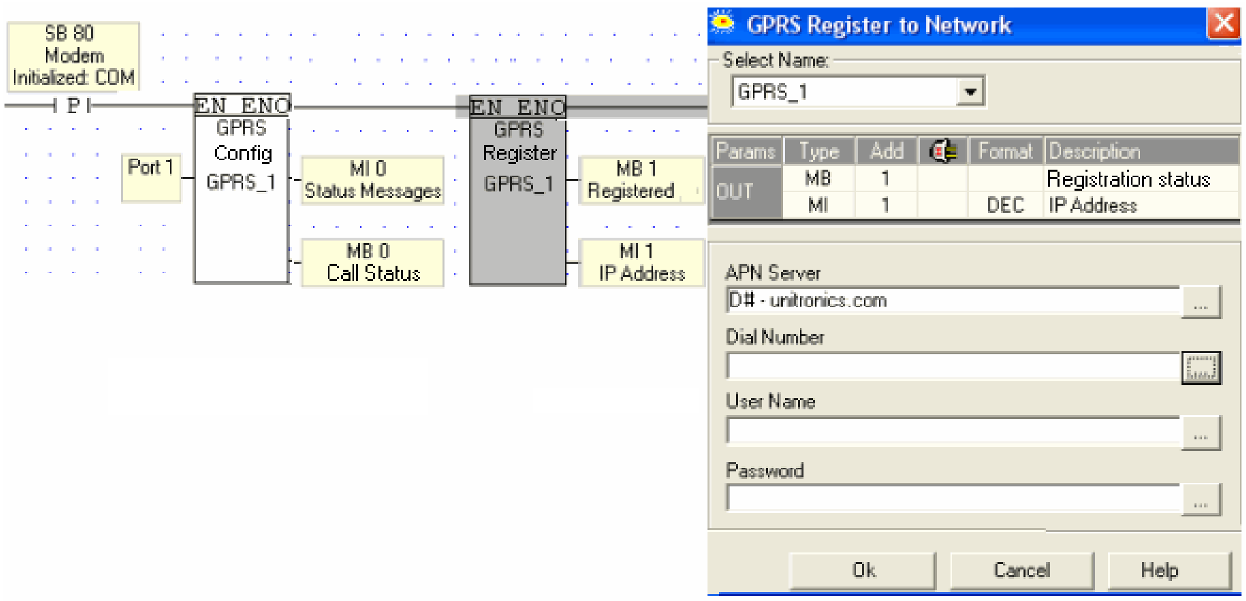
Name: GPRS_1

Params	Type	Add			Format	Description
IN	D#		0		DEC	Port 1
OUT	MI	0			DEC	Status Messages
	MB	0				Call Status

Parameter	Type	Function
Name		The name of the Configuration.
Port Number	Constant	Click the drop-down arrows to view available ports; click the port you want to use.
Status Messages	MI	The value of the linked MI indicates GPRS status messages.
Call Status	MB	This bit turns ON when the remote device has been accessed and the GPRS connection is established.

GPRS Register to Network

This FB provides the parameters that enable the PLC to connect to the Internet via the GPRS cellular network.



Select Name: GPRS_1

Params	Type	Add		Format	Description
OUT	MB	1			Registration status
	MI	1		DEC	IP Address

APN Server: D# - unitronics.com

Dial Number:

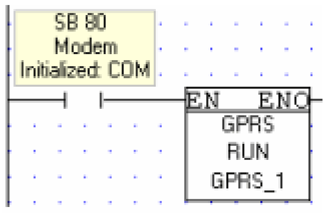
User Name:

Password:

Parameter	Type	Function
Name		Select the name of the GPRS Configuration that Connect will use to connect to the GPRS network.
Registration Status	MB	Turns ON when the PLC is assigned an IP address and registered by the GPRS network.
IP Address	MI	When the PLC registers on the GPRS network, it is assigned a dynamic IP address. This is the start of a vector that contains the IP address that is assigned to the modem when the modem registers with the GPRS network. The vector is 4 MIs long. The low byte of each MI provides the number for an octet within the IP address. If, for example, the IP address is linked to MI 0, and the low bytes of MI 0 to MI 3 contain the values 192, 198, 192, 45, the IP address will be 192.198.192. 45.
APN Server	Constant or MI	The name of the APN (Access Point Name) Server given by your GPRS service provider.
Dial Number (optional)	Constant or MI	These parameters are required by some GPRS service providers and GPRS modem manufacturers.
User Name (optional)	Constant or MI	
Password (optional)	Constant or MI	

GPRS Run

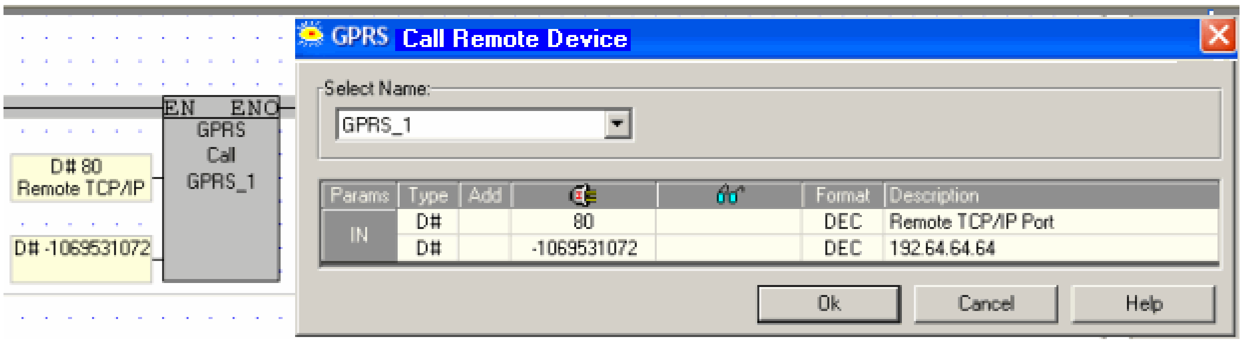
This element enables data communications via GPRS and must be included in all GPRS applications. The activating conditions should include the appropriate Modem Initialized SB for the COM Port.



Start Call, End Call

Start Call

This FB provides the Port and IP address of the remote device, enabling the PLC to call the remote device and establish a data communications link.

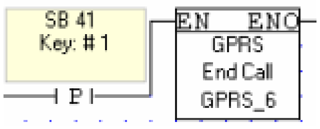


Params	Type	Add			Format	Description
IN	D#		80		DEC	Remote TCP/IP Port
	D#		-1069531072		DEC	192.64.64.64

Parameter	Type	Function
Name		Select the name of the GPRS Configuration that Connect will use to connect to the GPRS network.
Remote TCP/IP Port	Constant or MI	The access port of the remote device.
IP Address	Constant or MI	The address of the remote device.

End Call

This FB ends the current data communications sessions with a remote device, but does not terminate the connection to the GPRS network.

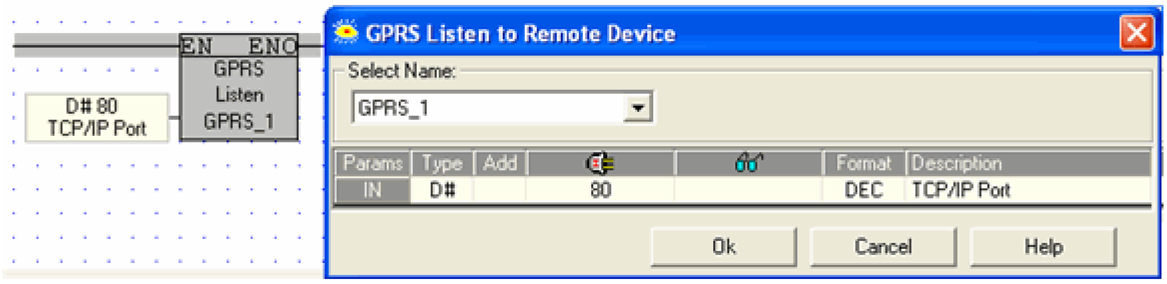


Note • Wavecom modems do not support the End Call function. In order to end a call, use the Unregister from Network FB, and then re-register to the network if required.

Listen /Stop Listening to Remote Device

Listen

This FB provides the Port and IP address of the remote device, enabling the PLC to call the remote device and establish a data communications link.

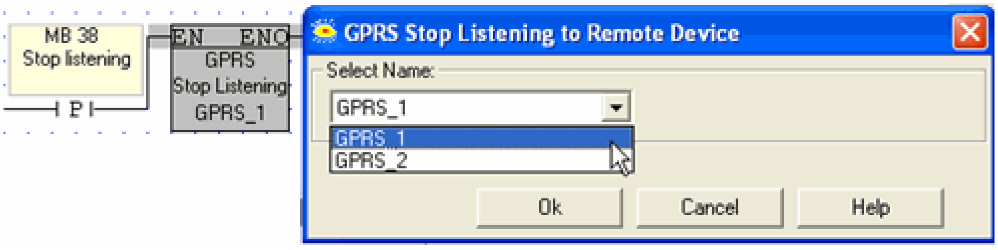


Params	Type	Add			Format	Description
IN	D#		80		DEC	TCP/IP Port

Parameter	Type	Function
Name		Select the name of the GPRS Configuration that Connect will use to connect to the GPRS network.
Remote TCP/IP Port	Constant or MI	The port the PLC uses to listen to the remote device.

Stop Listening

This FB ends the current data communications sessions with a remote device, but does not terminate the connection to the GPRS network.

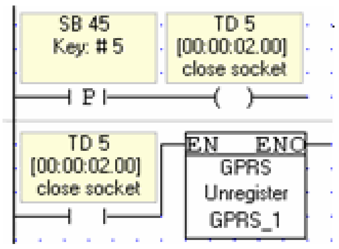


Parameter	Type	Function
Name		Select the name of the GPRS Configuration that Connect will use to connect to the GPRS network.

Note • Wavecom modems do not support the End Call function. In order to end a call, use the Unregister from Network FB, and then re-register to the network if required.

Unregister from Network

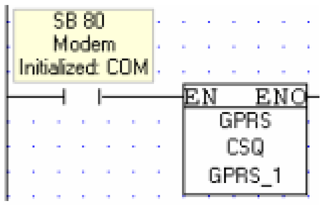
Use this to disconnect the PLC from the GPRS network, including a delay of a second or two to allow the socket time to close when de-registering.



Check Signal Quality

The signal quality value is placed in SI 185 GSM Signal Quality.

A value of -1(FFFF)signifies a modem error. This may be due to a weak signal; try repositioning the antenna. If this has no effect, check the modem.



GPRS Operands & Status Messages

Operands linked to GPRS FBs should be assigned Power-up Values; bits should be reset, and registers initialized to 0.

GPRS: Configuration FB

Call Status GPRS	**MB**	**Turns ON when:** Call successfully made	**Turns OFF when** Call is terminated
Status Messages Initialized to 0 when **GPRS: Configuration** is activated.	**MI**	Value Message 0 No message 1 GPRS network registration is in progress 2 Registration Complete: the modem successfully registered with and received an IP address from the GPRS network 3 Connected to GPRS network 4 Listen Mode: Initialization Begins 5 Listen Mode: Initialized 6 Listen Mode: Initialization Re-started 10 Start Call: Begins 11 Start Call: Complete 15 End call: Begins 16 End call: Complete 20 Unregistration Begins (Not during Call) 21 Unregistration Begins (During Call) 22 Unregistration Complete: the modem has successfully disconnected from the GPRS network 23 Listen Mode: Close process begins 24 Listen Mode: Closed 30 Check Signal Quality: Start 31 Check Signal Quality: End 40 Command not supported (When End Call is activated in an application using a Wavecom modem) 50 Modem reply error 51 Modem Timeout error 52 Network Registration: failled	

GPRS: Register to Network

Registration Status	**MB**	**Turns ON when:** The modem successfully registers with and received an IP address from the GPRS network	**Turns OFF when** The modem disconnects from the network
IP Address	**MI**	This is the start of a vector that contains the IP address that is assigned to the modem when the modem registers with the GPRS network. The vector is 4 MIs long. The low byte of each MI provides the number for an octet within the IP address. If, for example, the IP address is linked to MI 0, and the low bytes of MI 0 to MI 3 contain the values 192, 198, 192, 45, the IP address will be 192.198.192. 45.	

GPRS System Operands

SI	Description	Value
185	GSM Signal Quality	The value is written during COM Init of the GSM modem. The value is updated whenever the user uses the GSM Signal Quality FB. A value of -1 (FFFF)signifies a modem error. This may be due to a weak signal; try repositioning the antenna. If this has no effect, check the modem.

SBs 120-125 register the signals that each port receives from the DTR and DSR pins of a serial communication cable.
The DTR SBs 120, 122, and 124 are also used by the OS to control the DTR signal during RS485 serial communications, and during GPRS communications using the Sony Ericsson GPRS modem.

SB#	Description	Turns ON when:	Turns OFF when:	Reset by:
SB100	GPRS modem connected	Call Remote device begins GPRS incoming call is answered	End Session succeeds Disconnect from Network succeeds	OS
SB 120	DTR COM Port 1 (signal output from PLC)	DTR signal present	DTR signal absent	OS, may also be reset by user
SB 121	DSR COM Port 1 (signal input to PLC)	DSR signal present	DSR signal absent	OS
SB 122	DTR COM Port 2 (signal output from PLC)	DTR signal present	DTR signal absent	OS, may also be reset by user
SB 123	DSR COM Port 2 (signal input to PLC)	DSR signal present	DSR signal absent	OS
SB 124	DTR COM Port 3 (signal output from PLC)	DTR signal present	DTR signal absent	OS, may also be reset by user
SB 125	DSR COM Port 3 (signal input to PLC)	DSR signal present	DSR signal absent	OS

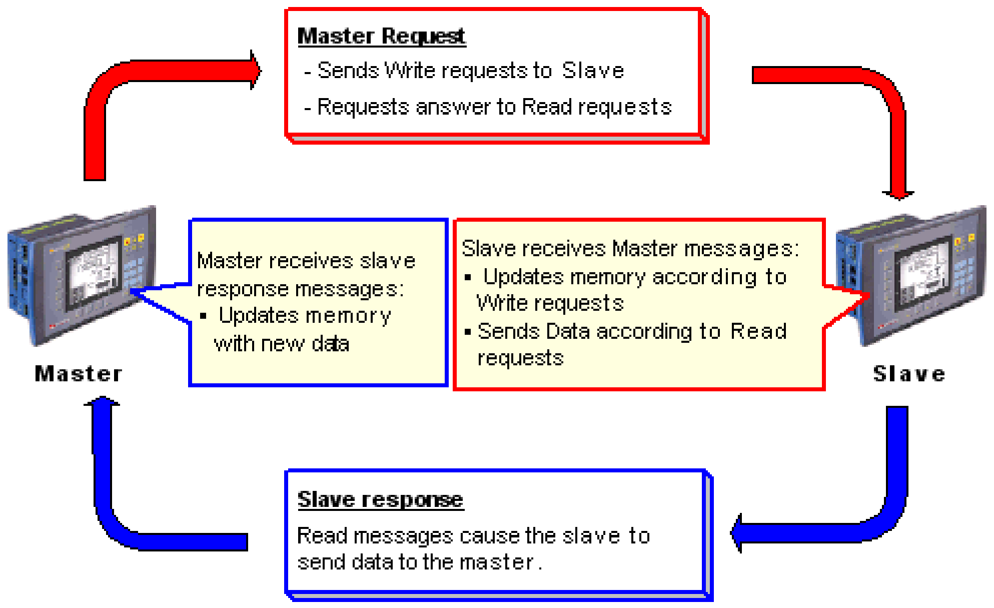
PLC DataCom

PLC DataCom Overview

If your controller comprises an Ethernet card, you can use Remote PLC DataCom commands to communicate mixed data messages, containing both register and bit values, to remote Unitronics PLCs over TCP/IP.

Any controller in the network may function as either master or slave via the controller's Ethernet port.

Remote PLC DataCom runs over UDP. TCP is not supported. Specific information on implementing Ethernet is provided in the topic Using Ethernet.


Master Request
- Sends Write requests to Slave
- Requests answer to Read requests
Master receives slave response messages:
▪ Updates memory with new data
Slave receives Master messages:
▪ Updates memory according to Write requests
▪ Sends Data according to Read requests
Master
Slave
Slave response
Read messages cause the slave to send data to the master.

Using Remote PLC DataCom

Before using a Remote PLC DataCom operation in your application, you must:

- Assign IP addresses to both master and slave PLCs. This is done by placing TCP/IP Card Init FBs in the ladder application of both master and slave.
- Determine which socket will be used for PLC DataCom.
  - Socket 0: This socket is set to UDP broadcast by default.
  If you select Socket 0, you must set it to support UDP Unicast (device to device) by turning SB 159 OFF. You must also include a TCP/IP Socket Init, set to UDP mode.
  - Sockets 1-3: These are set to TCP by default.
  In order to use these sockets, use a TCP/IP Socket Init function to switch the socket to UDP mode.
  If you set Sockets 1-3 to UDP, they will be in Broadcast mode.
- Include at least 1 Remote PLC DataCom Configuration FB in the ladder application of **both** master and slave.
- The condition that activates the Configuration must turn ON for a single program scan (positive transition recommended).
  However, the Remote PLC DataCom configuration must be scanned

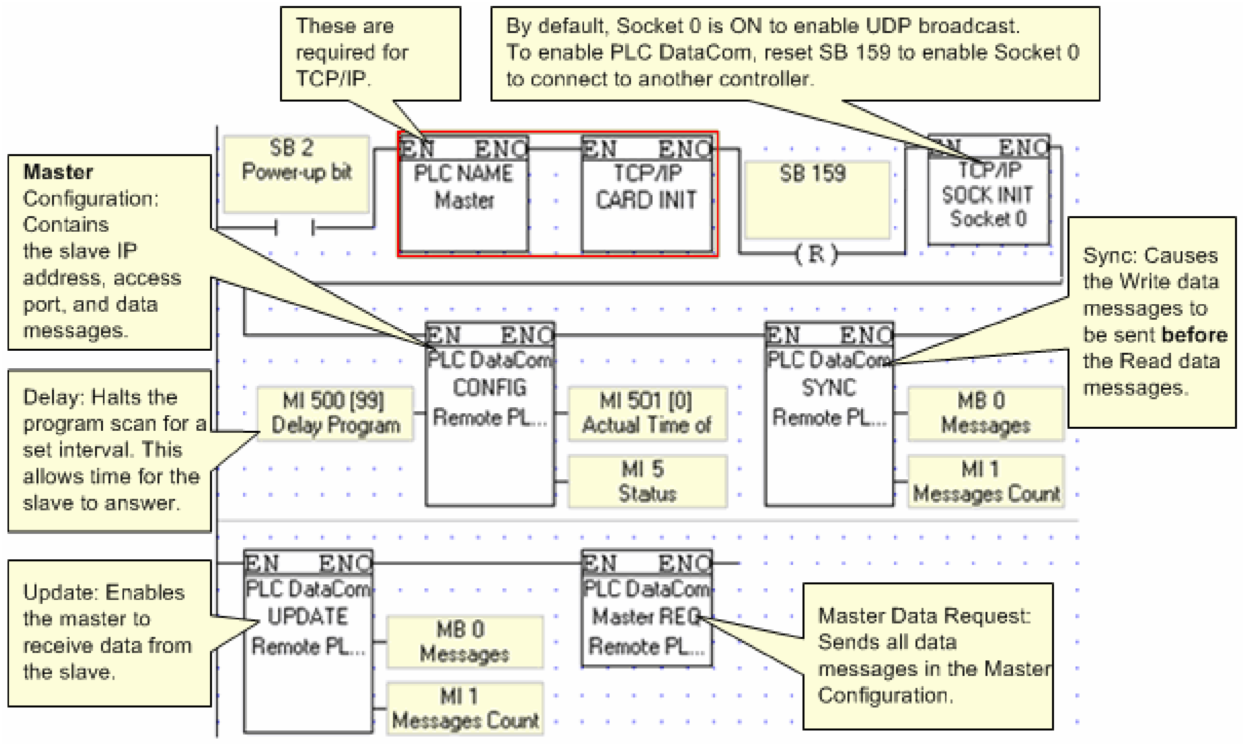
during every program cycle--after the Configuration is activated. One way to ensure this is by placing the configuration in the first subroutine of the main module.

- Enable data transfer by including an Update FB in the ladder application of **both** master and slave.

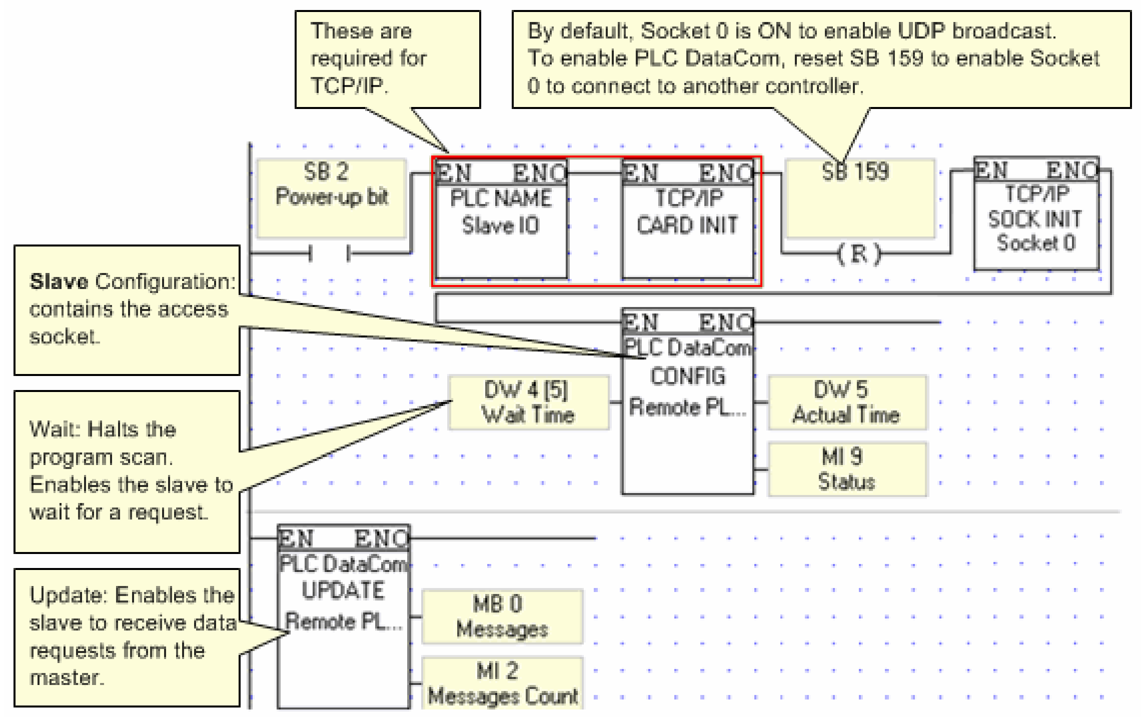
You can also include a DataCom Data Synchronization function in your master's ladder application. This causes the Write messages to be sent before the Read messages.

The programs below show how to implement PLC DataCom via Socket 0.

Master Program


These are required for TCP/IP.
By default, Socket 0 is ON to enable UDP broadcast. To enable PLC DataCom, reset SB 159 to enable Socket 0 to connect to another controller.
SB 2 Power-up bit
EN ENO PLC NAME Master
EN ENO TCP/IP CARD INIT
SB 159
(R)
EN ENO TCP/IP SOCK INIT Socket 0
Master Configuration: Contains the slave IP address, access port, and data messages.
Sync: Causes the Write data messages to be sent **before** the Read data messages.
EN ENO PLC DataCom CONFIG Remote PL...
MI 500 [99] Delay Program
MI 501 [0] Actual Time of
MI 5 Status
EN ENO PLC DataCom SYNC Remote PL...
MB 0 Messages
MI 1 Messages Count
Delay: Halts the program scan for a set interval. This allows time for the slave to answer.
Update: Enables the master to receive data from the slave.
EN ENO PLC DataCom UPDATE Remote PL...
MB 0 Messages
MI 1 Messages Count
EN ENO PLC DataCom Master REQ Remote PL...
Master Data Request: Sends all data messages in the Master Configuration.

Slave Program


These are required for TCP/IP.
By default, Socket 0 is ON to enable UDP broadcast. To enable PLC DataCom, reset SB 159 to enable Socket 0 to connect to another controller.
SB 2 Power-up bit
EN ENO PLC NAME Slave IO
EN ENO TCP/IP CARD INIT
SB 159
(R)
EN ENO TCP/IP SOCK INIT Socket 0
Slave Configuration: contains the access socket.
EN ENO PLC DataCom CONFIG Remote PL...
Wait: Halts the program scan. Enables the slave to wait for a request.
DW 4 [5] Wait Time
DW 5 Actual Time
MI 9 Status
Update: Enables the slave to receive data requests from the master.
EN ENO PLC DataCom UPDATE Remote PL...
MB 0 Messages
MI 2 Messages Count

Note • Non-Legacy controllers such as the V570 have a fast program scan. In order to ensure that operations work correctly, ensure that the program scan is greater than 3 msec. This can be done by increasing the scan time with the Idle utility.

FB Operations

Remote PLC DataCom Operations are located on the FBs menu.

Configuration

DataCom Data Synchronization

Update

Master Data Request

PLC DataCom: Configuration

A PLC DataCom Configuration FB must be included in both master and slave Ladder applications as shown below. PLC DataCom Operations are located on the FBs menu.

Master Configuration

The Master Configuration enables you to send data requests to Slave PLCs.

To display the Master parameters, select Master under Master/Slave.

Each Configuration can contain both Read and Write Mix Requests. Each request may be for a different data type. Your data request must include :

- Master and Slave operand addresses
- Length of vector
- Direction: Read or Write

After you add a request, the OK button is disabled. Click the Compile button to see current buffer status; if the buffer contains less than the maximum number of bytes, the OK is enabled.

Read Write Limitations

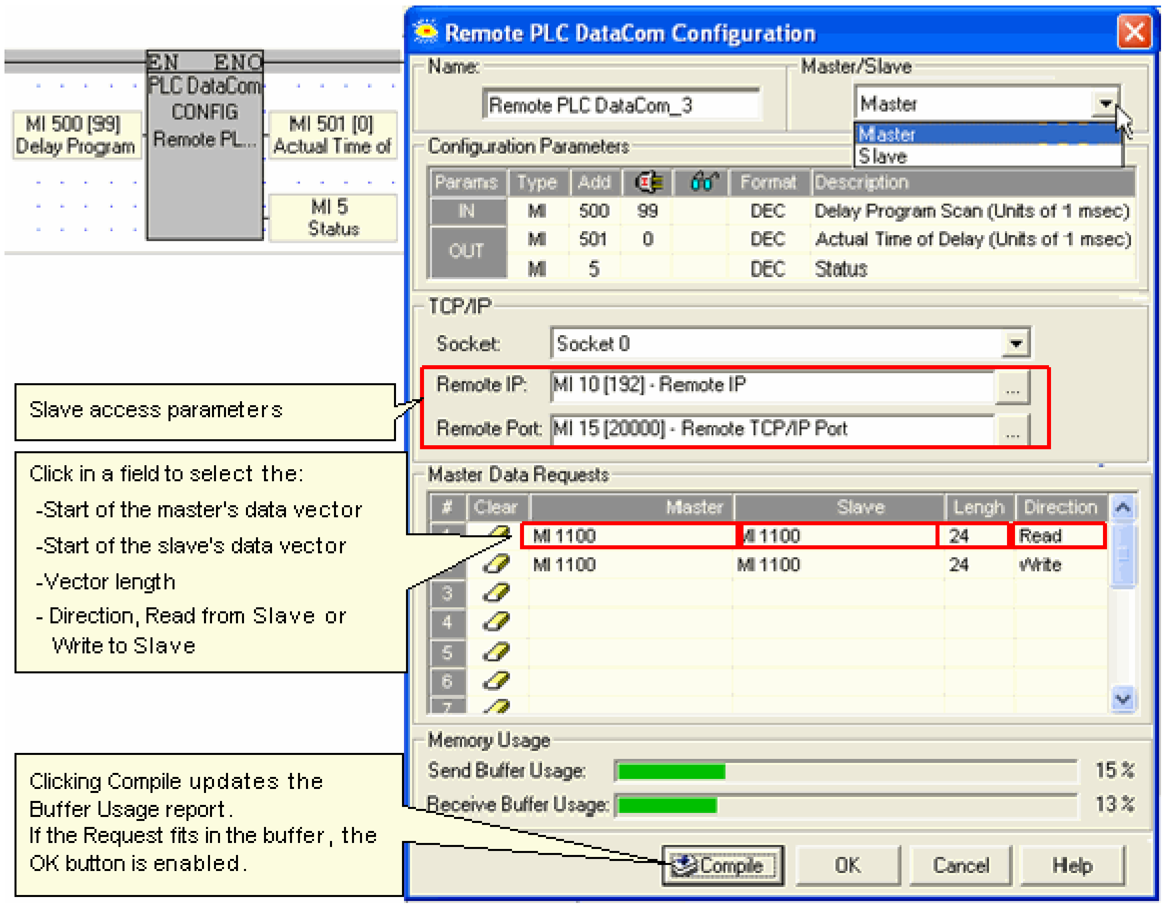
Only the following data types may be used in Read/Write requests: MI, ML, DW, MB, I and O.

Registers: may only be read/written to the same data type.

Booleans: Inputs cannot be written to.

Booleans, Read Write		
I	→	**MB, O**
O	↔	**MB, O**
MB	↔	**MB, O**

Registers, Read Write		
MI	↔	**MI**
ML	↔	**ML**
DW	↔	**DW**



Remote PLC DataCom Configuration

Name: Remote PLC DataCom_3

Master/Slave: Master (Master, Slave)

Configuration Parameters

Params	Type	Add			Format	Description
IN	MI	500	99		DEC	Delay Program Scan (Units of 1 msec)
OUT	MI	501	0		DEC	Actual Time of Delay (Units of 1 msec)
	MI	5			DEC	Status

TCP/IP

Socket: Socket 0

Remote IP: MI 10 [192] - Remote IP

Remote Port: MI 15 [20000] - Remote TCP/IP Port

Master Data Requests

#	Clear	Master	Slave	Lengh	Direction
1		MI 1100	MI 1100	24	Read
2		MI 1100	MI 1100	24	Write
3					
4					
5					
6					
7					

Memory Usage

Send Buffer Usage: 15 %

Receive Buffer Usage: 13 %

Compile OK Cancel Help

Slave access parameters

Click in a field to select the:
- Start of the master's data vector
- Start of the slave's data vector
- Vector length
- Direction, Read from Slave or Write to Slave

Clicking Compile updates the Buffer Usage report.
If the Request fits in the buffer, the OK button is enabled.

Parameter	Type	Function
Configuration parameters		
Delay Program Scan	MI	This is the amount of time a Master PLC will halt the program scan in order to wait for an answer from a Slave. Time out units are defined in 1 msecs; the maximum is 100 msec.
Actual Time of Delay	MI	This is the amount of time the Master PLC actually waited for a Slave response.
Status	MI	The value in the Status MI indicates the following: 0 - No Error 1 - Master: waiting for message (The Configuration parameter Delay Program Scan time has not been exceeded) Errors: 3 - The number of Read requests or the number of Write requests is greater than 16. 4 - The Master message length or the Slave message length exceeds the legal limit of 500 bytes. 5 - No Ethernet card can be found. 6 - The Ethernet card is not set to UDP. 7 - The Wait Time has been exceeded (Wait time > 100 in the Master or Wait time > 10 in Slave). 8 - Master has not received message (The Configuration parameter Delay Program Scan time has been exceeded). 9 - PLC or network error. 10 - The remote Slave IP does not exist. 11 - Checksum Error in the message received by the Master.

TCP/IP

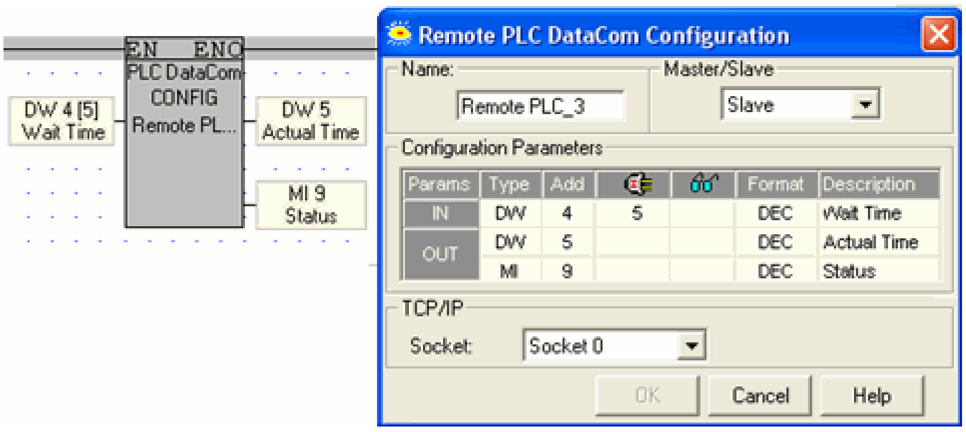
Socket		In order to support Remote PLC DataCom, the socket you use must be set to UDP, Multicast mode. By default, Socket 0 is set to UDP, Unicast. To enable PLC DataCom, reset SB 159 to change Socket 0 to Multicast.
Remote IP	MI	The IP address of the remote Slave unit. Note • The IP vector is 4 MIs long. The low byte of each MI provides the number for an octet within the IP address. If, for example, the IP address is linked to MI 0, and the low bytes of MI 0 to MI 3 contain the values 192, 198, 192, 45, the IP address will be 192.198.192. 45.
Remote Port	MI	The access port of the remote Slave unit.

Master Data Requests
Click in a field to select the:
-Start of the master's data vector
-Start of the slave's data vector
-Vector length
- Direction, Read from Slave or Write to Slave

Memory usage
The function uses two buffers, Send and Receive. Each buffer can contain a maximum of 500 bytes.
Clicking Compile updates the Buffer Usage report.
If the Request fits in the buffer, the OK button is enabled.

Slave Configuration

The Slave Configuration enables the Slave PLC to receive data requests.


EN ENO
PLC DataCom CONFIG Remote PL...
DW 4 [5] Wait Time
DW 5 Actual Time
MI 9 Status
Remote PLC DataCom Configuration
OK
Cancel
Help

Name: Remote PLC_3
Master/Slave: Slave

Configuration Parameters

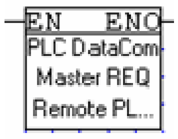
Params	Type	Add			Format	Description
IN	DW	4	5		DEC	Wait Time
OUT	DW	5			DEC	Actual Time
	MI	9			DEC	Status

TCP/IP
Socket: Socket 0

Parameter	Type	Function
Configuration parameters		
Delay Program Scan	MI	This is the amount of time a Slave PLC will halt the program scan in order to wait for a Master message. Time out units are defined in 1 msecs; the maximum is 10 msec.
Actual Time of Delay	MI	This is the amount of time the Slave PLC actually waited for a Master message.
Status	MI	The value in the Status MI indicates the following: 0 - No Error 1 - Master: waiting for message (The Configuration parameter Delay Program Scan time has not been exceeded) Errors: 3 - The number of Read requests or the number of Write requests is greater than 16. 4 - The Master message length or the Slave message length exceeds the legal limit of 500 bytes. 5 - No Ethernet card can be found. 6 - The Ethernet card is not set to UDP. 7 - The Wait Time has been exceeded (Wait time > 100 in the Master or Wait time > 10 in Slave). 8 - Master has not received message (The Configuration parameter Delay Program Scan time has been exceeded). 9 - PLC or network error. 10 - The remote Slave IP does not exist. 11 - Checksum Error in the message received by the Master.

Master Data Request

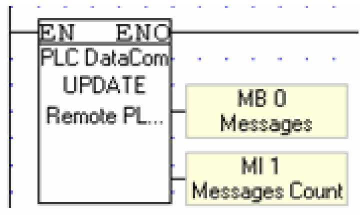
Activate a Master Data Request to send all data messages in the Master Configuration.



Update

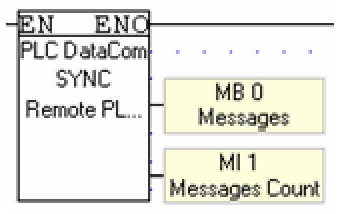
Update enables a PLC to receive Remote PLC DataCom messages. Update must be included in Slave ladder applications to enable a Slave to receive messages, and must be included in the Master ladder application to enable the Master to receive responses from slave PLCs.

Update should be placed on the left-hand ladder rail in the Main routine.



DataCom Data Synchronization

This function causes the Write data messages to be sent before the Read data messages.



PLC DataCom Status Messages

The value in the Configuration's Status MI indicates the following:

0 - No Error

1 - Master: waiting for message (the Configuration parameter Delay Program Scan time has not been exceeded)

3 - The number of Read requests or the number of Write requests is greater than 16.

4 - The Master message length or the Slave message length exceeds the legal limit of 500 bytes.

5 - No Ethernet card can be found.

6 - The Ethernet card is not set to UDP.

7 - The Wait Time has been exceeded (Wait time > 100 in the Master or Wait time > 10 in Slave).

8 - Master: waiting for message (the Configuration parameter Delay Program Scan time has been exceeded).

9 - PLC or network error.

10 - The remote Slave IP does not exist.

11 - Checksum Error in the message received by the Master.

Note •	To automatically reconnect a lost Ethernet connection, turn SB 168 Automatically reconnect (keep alive) ON at power-up.
•	To enable PLC to disconnect when the Ethernet connection is inactive for a period of time, assign timeout values in SIs 103 -106
•	To enable a PLC attempt to reconnect when there is no communication from the connected device for a period of time, assign timeout values in SIs 107 -110

Protocol FB, Serial

Protocol FB (Serial) Overview

Use the Communication Protocol operations to enable your controller to exchange data with external devices, such as frequency converters, bar-code readers, and printers via a Vision COM port. Protocol operations are located on the FBs menu.

How the Protocol FB communicates data between Vision controllers and other devices

A device, such as a magnetic card reader, may use its own proprietary protocol. If you know the protocol's structure, you can use the Protocol FB to structure messages accordingly. This enables a Vision controller to exchange data with the device using the device's own protocol.

Note that before you can use Protocol operations, you must configure the COM port to be connected to the external device in accordance with the device's requirements. This is done by placing a COM Port Init FB in your Ladder application.

Basic Example

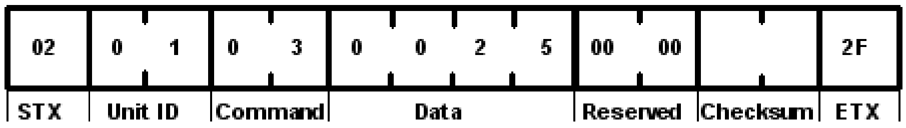
The example below shows how a Vision controller can, via COM port 2, read Register 25 within a device that has the ID number of 01. The device uses the simple protocol shown below.

Device Protocol

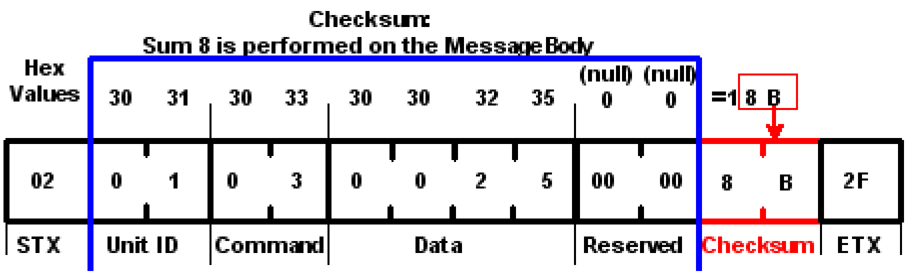
Parameter name	**Length (bytes)**	**Value**	**Comments**
STX	1	STX 02(hex)	All messages must begin with the STX character
Unit ID	2	ID# range = 0-99	All messages must contain the ID number of the device that the controller is communicating with
Command	2	03=Read Register 04=Read Bit 05=Write Register	Commands are embedded in Data Request messages.
Data (Request)	4		Vision to Device
Data (Response)	8		Device to Vision
Reserved	2	2 Null characters=00,00	These characters must be included in all messages.
Checksum	2	Sum 8	
ETX	1	/ backslash) 2F(hex)	All messages must end with the '/'character

Message Structure: Data Request (Vision to device)

The device can only receive request messages that are structured as shown below.

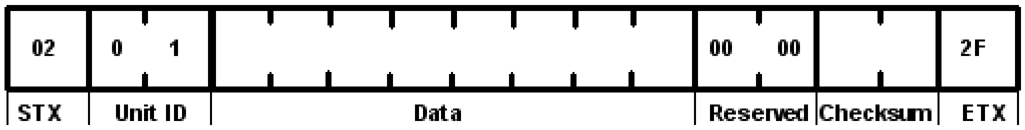


Checksum



Message Structure: Data Response (device to Vision)

The device sends data in messages that are structured as shown below.



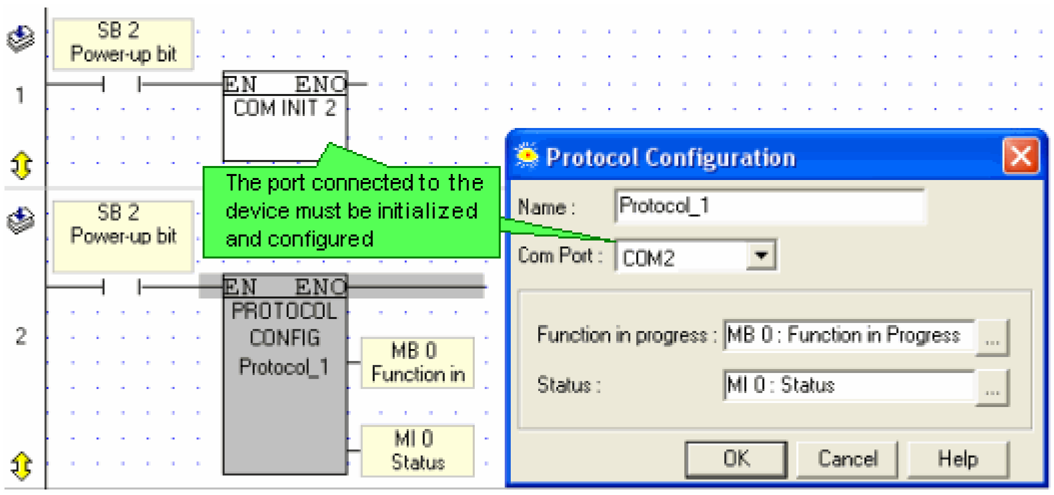
Protocol Ladder Requirements

In order to enable a Vision controller to request and receive data from a device, the controller's Ladder application must contain:

- A COM Init FB to initialize the serial port that will be linked to the device.
- A Protocol Configuration FB set to use the serial port.
- A Protocol Send FB, which enables the controller to send data request messages to the device.
- A Protocol Scan FB, which enables the controller to receive data response messages from the device. Note that in order to receive new messages, you **must clear the communication buffer after each message by using the Reset Buffer FB.**

COM Init and Configuration

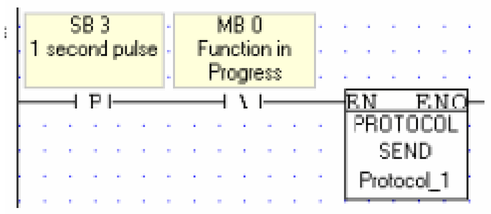
Since COM Port 2 is connected to the device, the application initializes and configures COM 2 at Power-up.

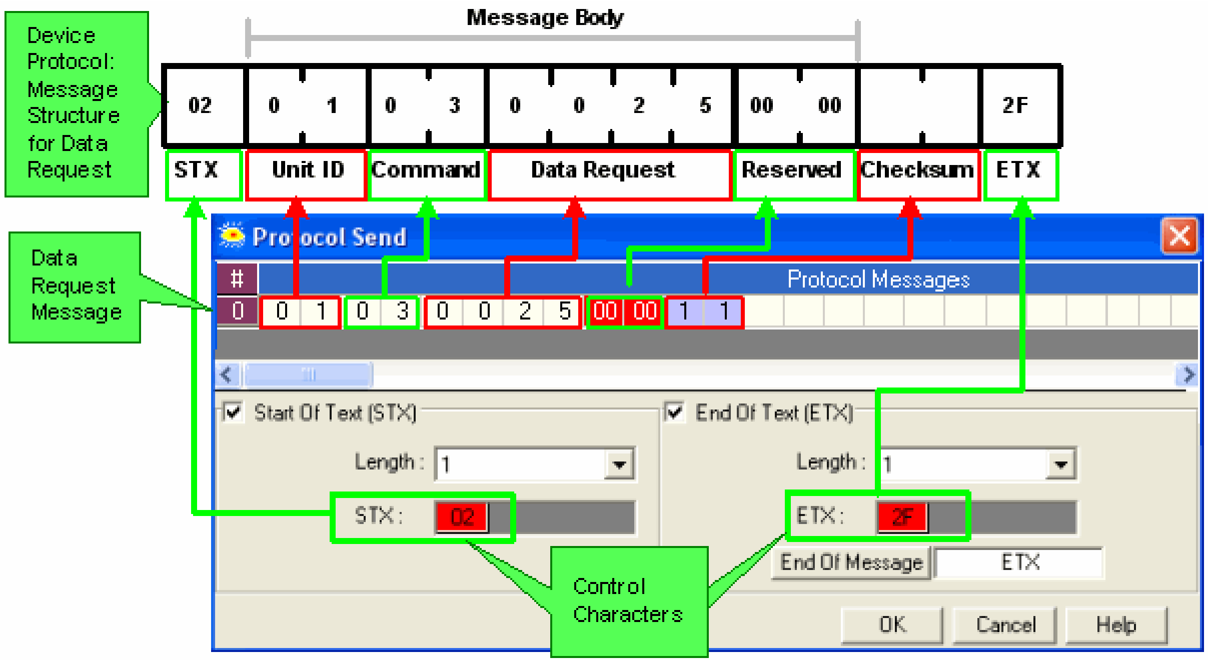


Name : Protocol_1
Com Port : COM2
Function in progress : MB 0 : Function in Progress
Status : MI 0 : Status

Send: Data Request

The application requests data via the Protocol Send FB. The Send FB enables you to structure messages that conform to any protocol's requirements.
 Below, the Vision sends a data request to Unit 01. The request is Command 03, Read Register, and the register is Register 25.

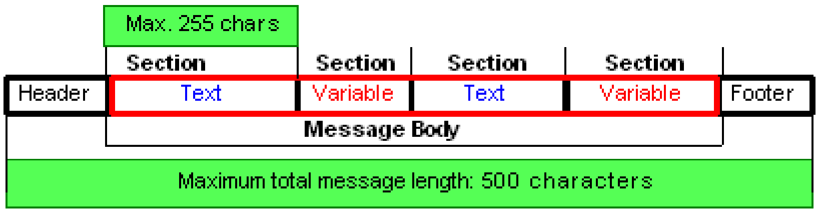




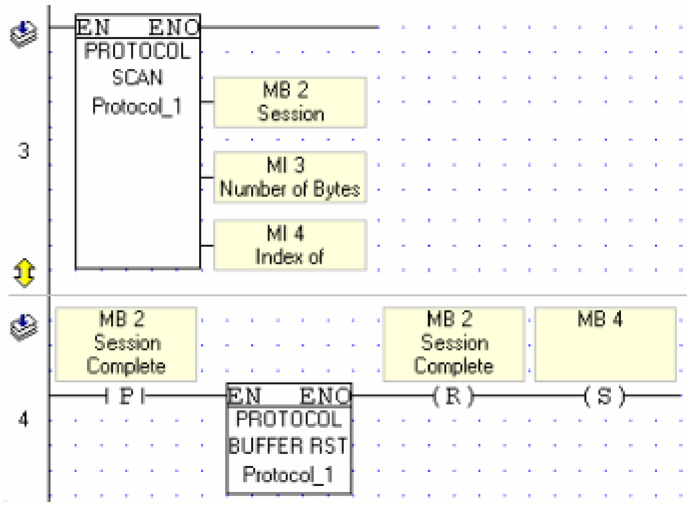
Start Of Text (STX) ✓ Length : 1 STX : 02
End Of Text (ETX) ✓ Length : 1 ETX : 2F
End Of Message ETX

Scan: Receiving Messages

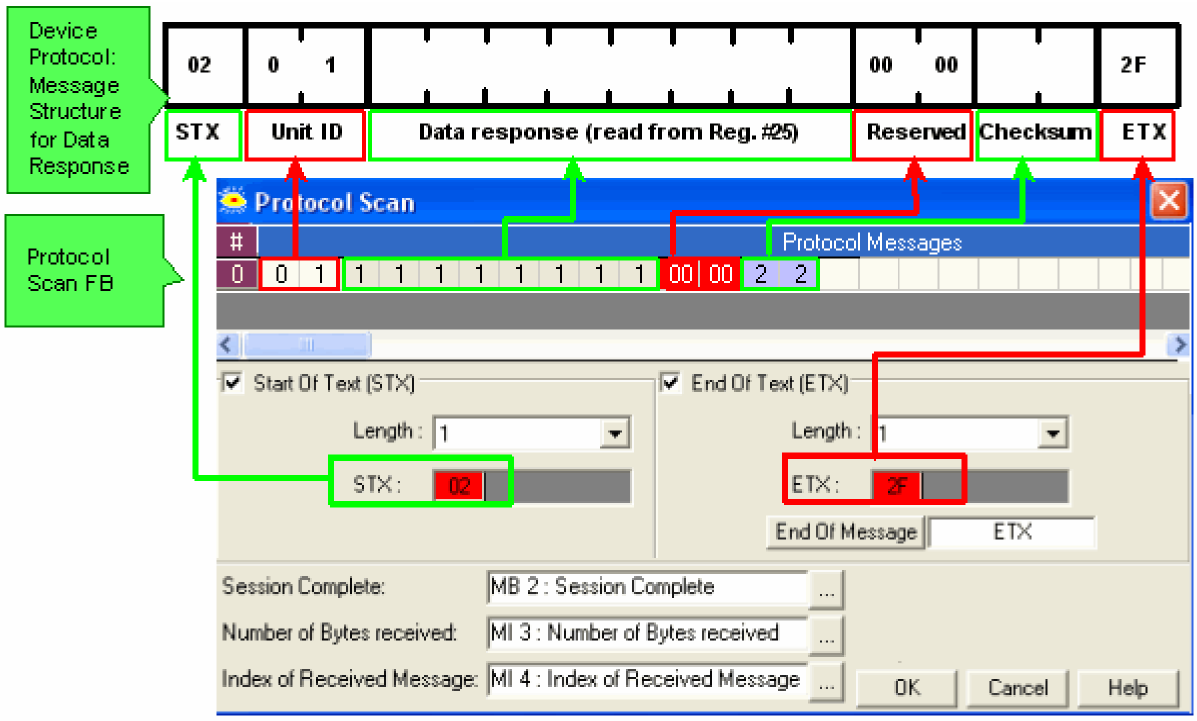
A Protocol Scan FB contains messages. Activating the Scan causes the Vision to check if it has received any of the messages contained within the Scan FB. Note that the controller can only receive messages that are defined in the Scan FB.

Note •	In order to receive new messages, you must clear the communication buffer by using the Reset Buffer operation. You can save the contents of the buffer at any time by using the Copy Buffer operation.
•	Maximum message length is 500 characters; a message may include up to 50 variables. Note that with in the body of a message, no section--whether text or variable--may exceed 255 characters. 

Below, the application enables the Vision to receive a response to Command 03. Note that Buffer reset follows the Scan.



Note that the Scan message contains a Data Response field that is long enough to contain the contents of a register.


Device Protocol: Message Structure for Data Response
Protocol Scan FB
02
0 1
00 00
2F
STX
Unit ID
Data response (read from Reg. #25)
Reserved
Checksum
ETX
Protocol Scan
#
Protocol Messages
0
0 1
1 1 1 1 1 1 1 1
00 00
2 2
Start Of Text (STX)
Length : 1
STX : 02
End Of Text (ETX)
Length : 1
ETX : 2F
End Of Message ETX
Session Complete: MB 2 : Session Complete
Number of Bytes received: MI 3 : Number of Bytes received
Index of Received Message: MI 4 : Index of Received Message
OK
Cancel
Help

For more information regarding Protocol operations, refer to the topics below.

FB Operations

Configuration

Scan

Send

Copy Buffer Contents

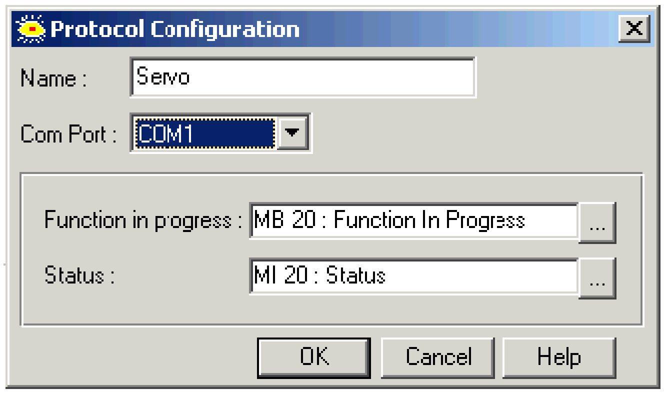
Reset Buffer

Note •	Reset Buffer must be used in the Ladder application to enable new messages to be received.
•	If you want to access a PLC via PC, remember that a PC accesses a PLC via a serial port. Vision controller ports are served by three communication buffers. Note that if all three buffers are busy processing communications, new requests are ignored until a buffer is free.

Protocol: Configuration

You create messages and attach variable data in Protocol Send and Scan operations. Each operation is linked to a Protocol Configuration. When a Send operation is called in your program, the Configuration determines which COM port is used for outgoing messages; when a Scan operation is activated, the PLC monitors that COM port for incoming messages.

Note •	The Configuration should be placed in the Main Routine, before any other FB operations. If the configuration is not active, Protocol operations will not be processed.
•	Before you place this FB in your Ladder, you must use a COM Port INIT FB to configure a COM port.



Name : Servo

Com Port : COM1

Function in progress : MB 20 : Function In Progress

Status : MI 20 : Status

Parameter		**Function**
Name		Identifies the configuration. You use the name to link Protocol Send and Scan operations to a configuration.
COM Port		Determines which port is used for communicating the configured protocol.
Function in Progress	MB	Link an MB; this MB will be ON when the Configuration is processing Scan or Send operations.
Status	MI	If errors occur when the Configuration is used for a Send or Scan operation, the value of the linked MI indicates which error has occurred.

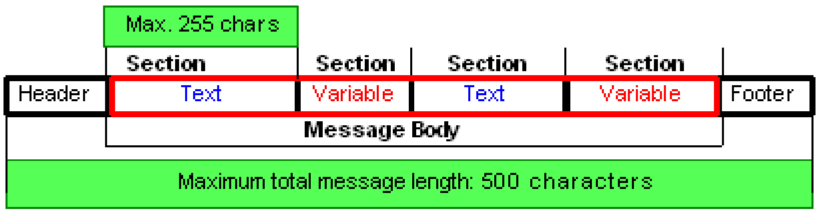
Protocol: Scan

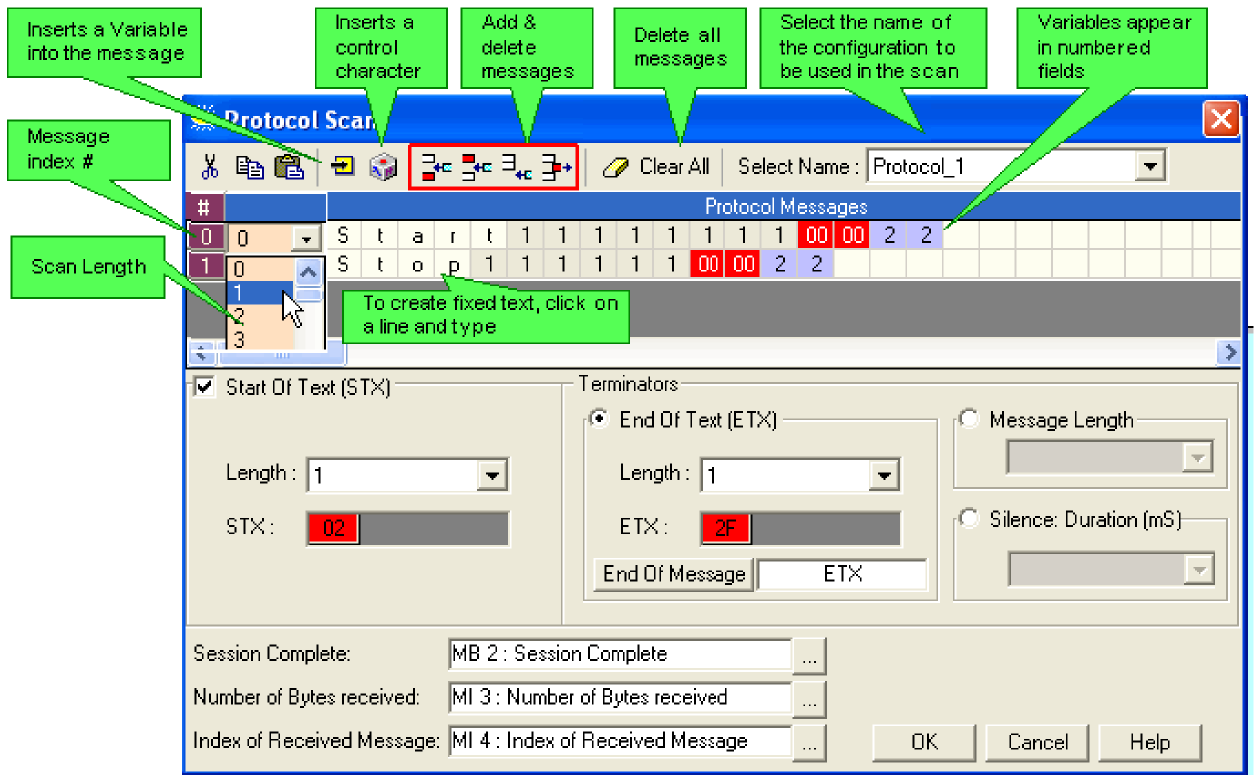
A Protocol Scan operation contains messages. Activating the Scan causes the PLC to check if it has received any of the messages contained within the Scan. The PLC can only receive messages that are contained in a Scan FB.

Each Protocol Scan operation is linked to a particular Protocol Configuration. The Scan only registers messages that are received via the COM port defined in that Configuration.

Scan is located under FBs>Protocol.

Note •	In order to receive new messages, you must clear the communication buffer by using the Reset Buffer operation. You can save the contents of the buffer at any time by using the Copy Buffer operation.
•	Maximum message length is 500 characters; a message may include up to 50 variables. Note that with in the body of a message, no section--whether text or variable--may exceed 255 characters.




Inserts a Variable into the message
Inserts a control character
Add & delete messages
Delete all messages
Select the name of the configuration to be used in the scan
Variables appear in numbered fields
Message index #
Scan Length
To create fixed text, click on a line and type
Protocol Scan
Clear All
Protocol Messages
#
Terminators
OK
Cancel
Help

Select Name : Protocol_1

```
0  0  S t a r t 1 1 1 1 1 1 1 1 00 00 2 2
1  0  S t o p 1 1 1 1 1 1 00 00 2 2
```

[✓] Start Of Text (STX)

Length : 1

STX : 02

Terminators

(●) End Of Text (ETX)

Length : 1

ETX : 2F

End Of Message: ETX

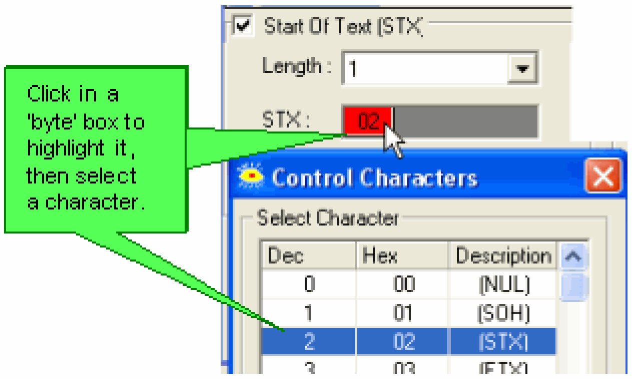
( ) Message Length

( ) Silence: Duration (mS)

Session Complete: MB 2 : Session Complete

Number of Bytes received: MI 3 : Number of Bytes received

Index of Received Message: MI 4 : Index of Received Message

Message index	Identifies the incoming message.
Scan Length	This enables you to define the length of each incoming message, via the drop-down box that follows Message index #. Scan Length causes the Scan FB to use the length of a message--including header and terminator--to define whether an incoming message is legal. The length of an incoming message cannot be less than the length specified in the Scan operation. **Note** • This parameter overrides Start Of Text and Terminator settings. If you use Scan Length, arrange the messages in ascending order, for example, message #0 may contain 3 bytes, #1 may contain 4 bytes, #2 may contain 6 bytes, etc.
Start of Text (STX)	Check Start Of Text to use an STX parameter to indicate the beginning of a message. Length: Enter the number of bytes required by the external device's protocol to mark STX. STX Character: **Click a 'byte' box and select an ASCII character**.  Click in a 'byte' box to highlight it, then select a character. Start Of Text (STX) Length : 1 STX : 02 Control Characters Select Character Dec Hex Description 0 00 (NUL) 1 01 (SOH) 2 02 (STX) 3 03 (ETX)
Terminators	Select one of the terminator parameters to indicate where the message ends. **Note** • Use Message Length only in cases where the messages are of equal length. If the messages are of different length, use a different terminator.

Parameter		**Function**	
Session Complete	MB	**Turns ON when:** Turns ON when the PLC receives a valid message, including the message's STX and terminator, after the Linked Configuration's Function in Progress MB turns OFF.	**Turns OFF when** Turns OFF whenever the Protocol Scan operation is called by the program. Turns OFF when a Reset Buffer operation runs. In your program, link the Session Complete bit to a positive transition contact, then use this condition to activate a Copy Buffer FB, as well as number of Bytes Received and Number of Received Message.
		Note • -If the PLC receives a message which is **not** defined in the Scan FB, the message is invalid. The Message Received MB remains OFF. -If a message in the Scan FB contains a Numeric Variable field', and the PLC receives this message with non-numerical characters in the field (except for leading spaces), the message is invalid. The Message Received MB remains OFF. -An example of leading spaces is if, for example, the Numeric Variable field is 4 boxes long, and the 'number' 22 is received. In this case, the Scan FB will register the number22 preceded by 2 leading spaces (_ _ 2 2).	
Number of Bytes Received	MI	This holds the number of bytes currently in the buffer. It is initialized by the system when a different message is received.	
Index of Received	MI	This holds the index number of the received message. It changes when another message enters the system.	

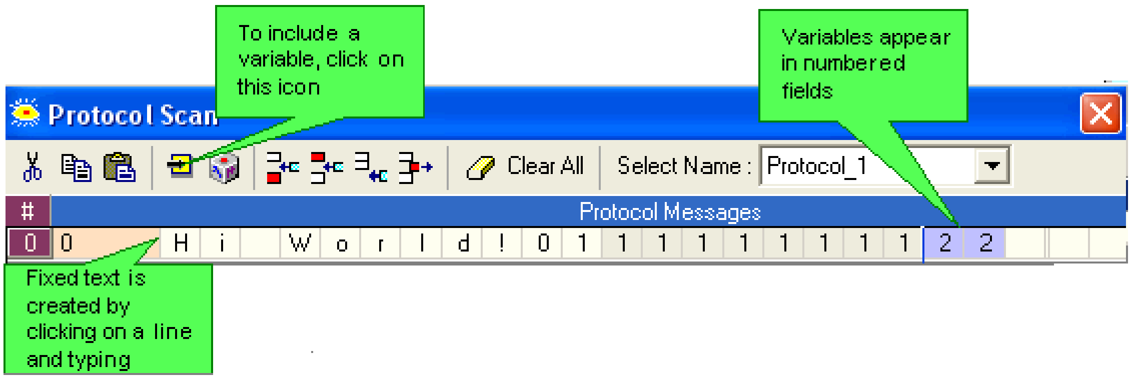
Message		The value in the linked MI is valid only when the Session Complete bit in ON. During this time, the linked MI contains the index number of the last valid incoming message. Once an invalid message has been received, the first incoming character of any message causes -1 to be written in the linked MI.

Protocol Messages

Messages can include:

- **Fixed Text**

This text does not change.

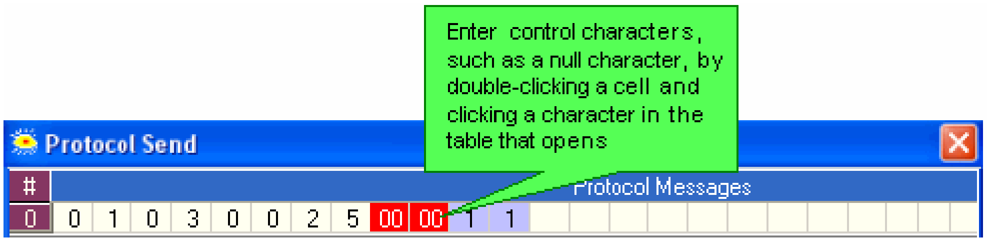

To include a variable, click on this icon
Variables appear in numbered fields
Protocol Scan
Clear All
Select Name : Protocol_1
Protocol Messages
0 0 H i W o r l d ! 0 1 1 1 1 1 1 1 1 1 2 2
Fixed text is created by clicking on a line and typing

- **Variables**

These variables may be received by the PLC: Numeric, Stream, and Checksum.

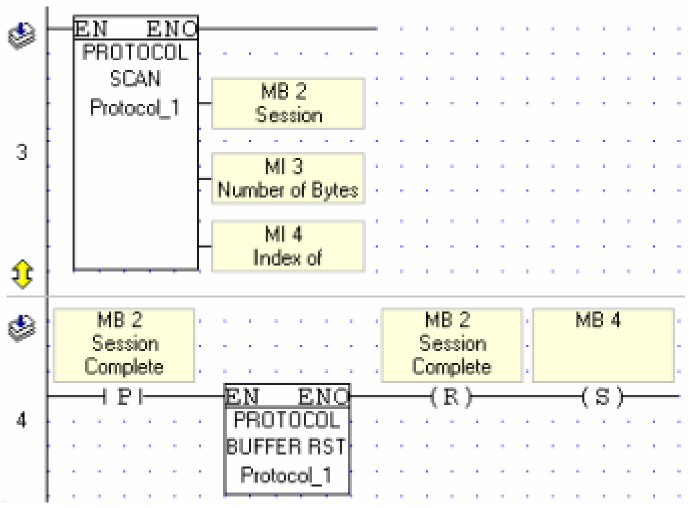
- **Control Characters**

These can be added according to protocol requirements.

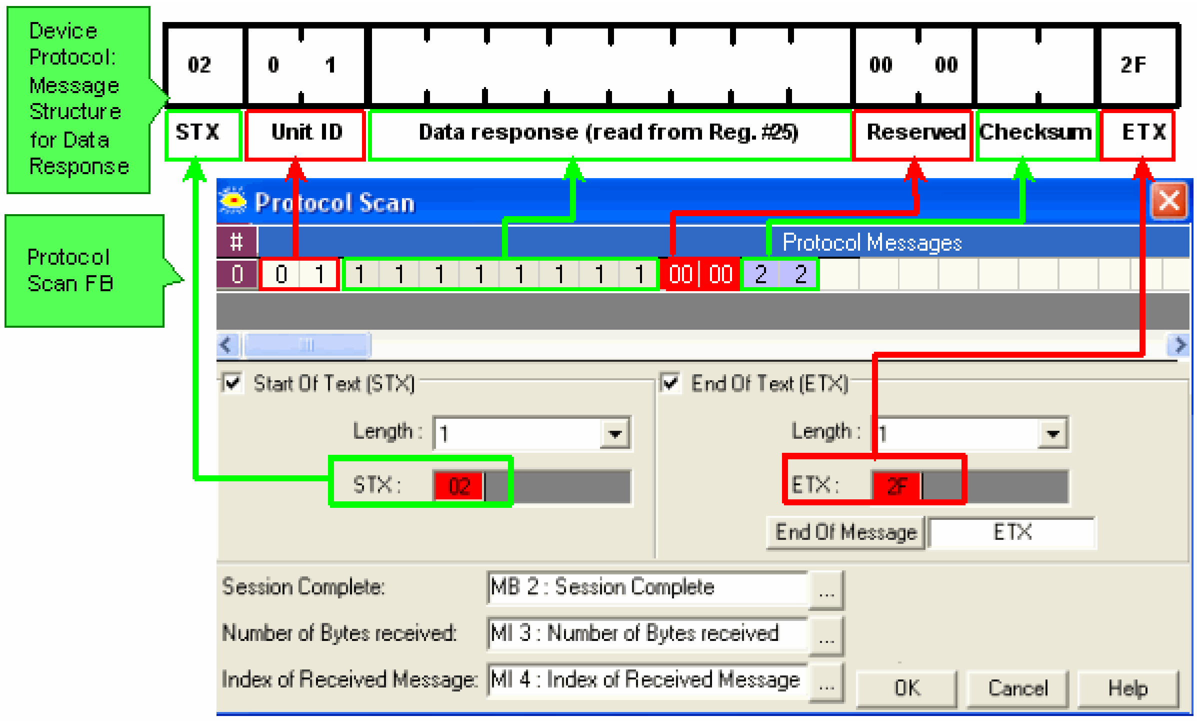

Enter control characters, such as a null character, by double-clicking a cell and clicking a character in the table that opens
Protocol Send
Protocol Messages
0 0 1 0 3 0 0 2 5 00 00 1 1

Example

Note that Buffer reset follows the Scan.



Note that the Scan message contains a Data Response field that is long enough to contain the contents of a register.



Start Of Text (STX) ✓
Length : 1
STX : 02

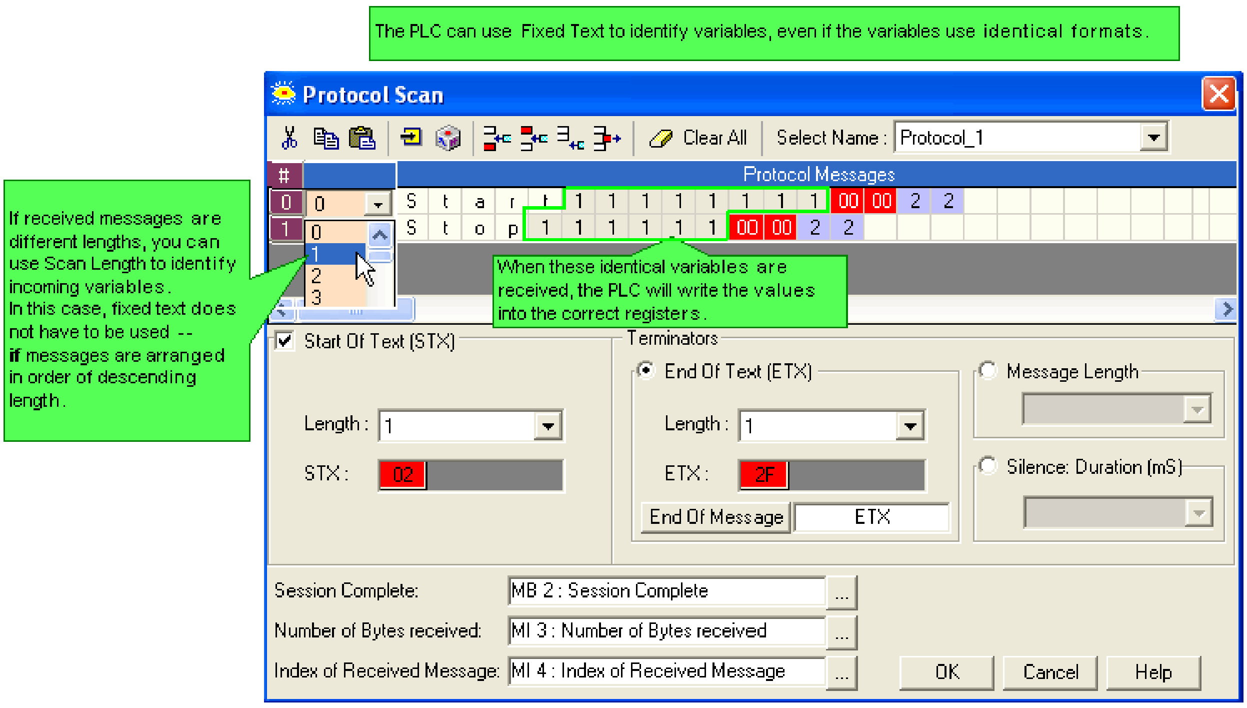
End Of Text (ETX) ✓
Length : 1
ETX : 2F
End Of Message: ETX

Session Complete: MB 2 : Session Complete
Number of Bytes received: MI 3 : Number of Bytes received
Index of Received Message: MI 4 : Index of Received Message

<u>Scan operations containing more than one message</u>

Note that if a Scan operation contains more than one message with identical variables that use the same format, the PLC will overwrite the variable values as they are received.

To enable the PLC to differentiate between the messages and their attached variables, include fixed text and/or control characters within the message.


The PLC can use Fixed Text to identify variables, even if the variables use identical formats.
If received messages are different lengths, you can use Scan Length to identify incoming variables. In this case, fixed text does not have to be used -- if messages are arranged in order of descending length.
When these identical variables are received, the PLC will write the values into the correct registers.
Protocol Scan
Clear All
Select Name : Protocol_1
#
Protocol Messages
0 0 S t a r t 1 1 1 1 1 1 1 1 00 00 2 2
1 0 S t o p 1 1 1 1 1 1 00 00 2 2
1
2
3

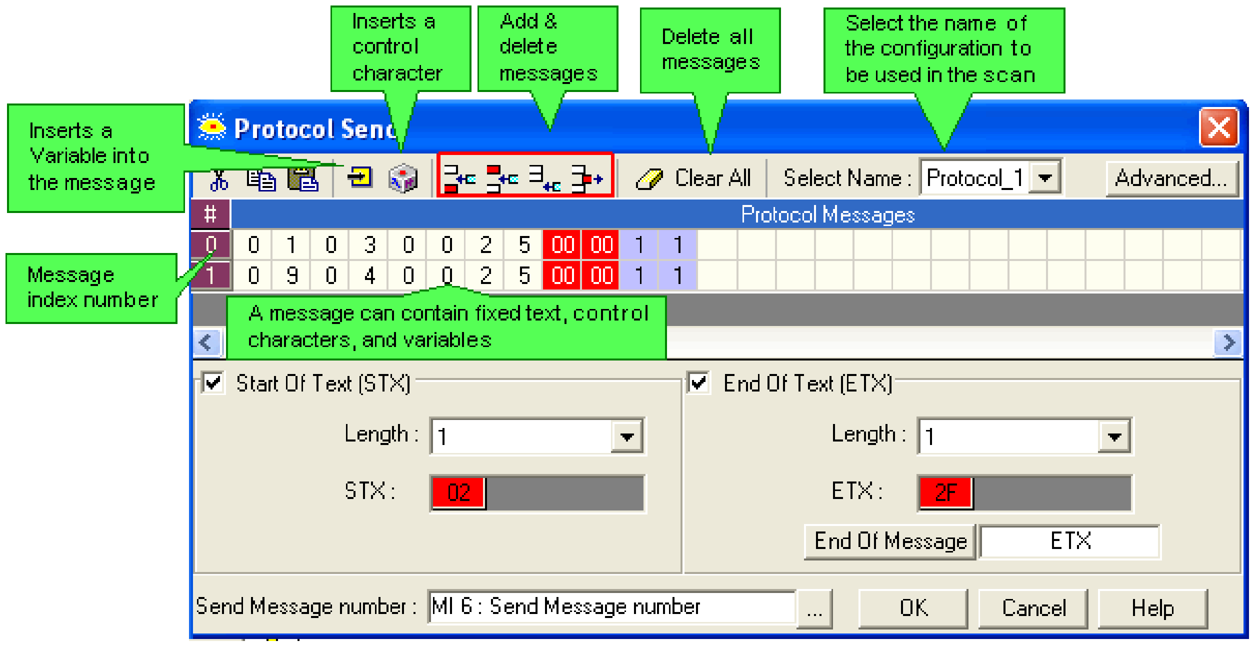
Start Of Text (STX) ✓
Length : 1
STX : 02

Terminators
(●) End Of Text (ETX)
Length : 1
ETX : 2F
End Of Message: ETX
( ) Message Length
( ) Silence: Duration (mS)

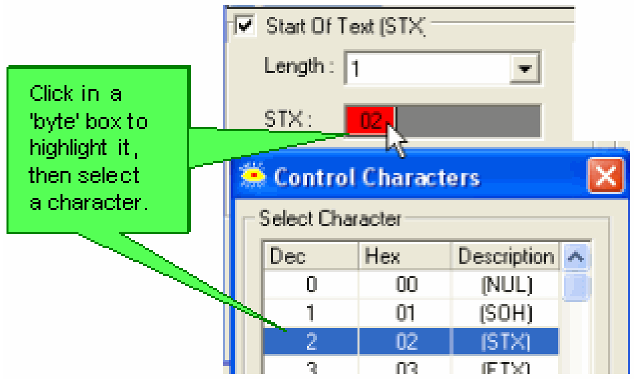
Session Complete: MB 2 : Session Complete
Number of Bytes received: MI 3 : Number of Bytes received
Index of Received Message: MI 4 : Index of Received Message

Protocol: Send

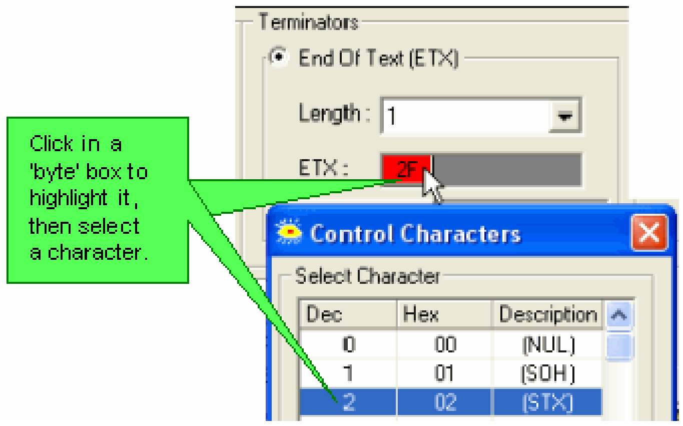
Each Protocol Send operation is linked to a particular Configuration; outgoing messages are sent via the COM port selected in that Configuration. Send is located on the FBs menu.


Inserts a control character
Add & delete messages
Delete all messages
Select the name of the configuration to be used in the scan
Inserts a Variable into the message
Message index number
A message can contain fixed text, control characters, and variables
Protocol Send
Clear All
Select Name : Protocol_1
Advanced...
Protocol Messages
Start Of Text (STX)
Length : 1
STX : 02
End Of Text (ETX)
Length : 1
ETX : 2F
End Of Message ETX
Send Message number : MI 6 : Send Message number
OK
Cancel
Help

#												
0	0	1	0	3	0	0	2	5	00	00	1	1
1	0	9	0	4	0	0	2	5	00	00	1	1

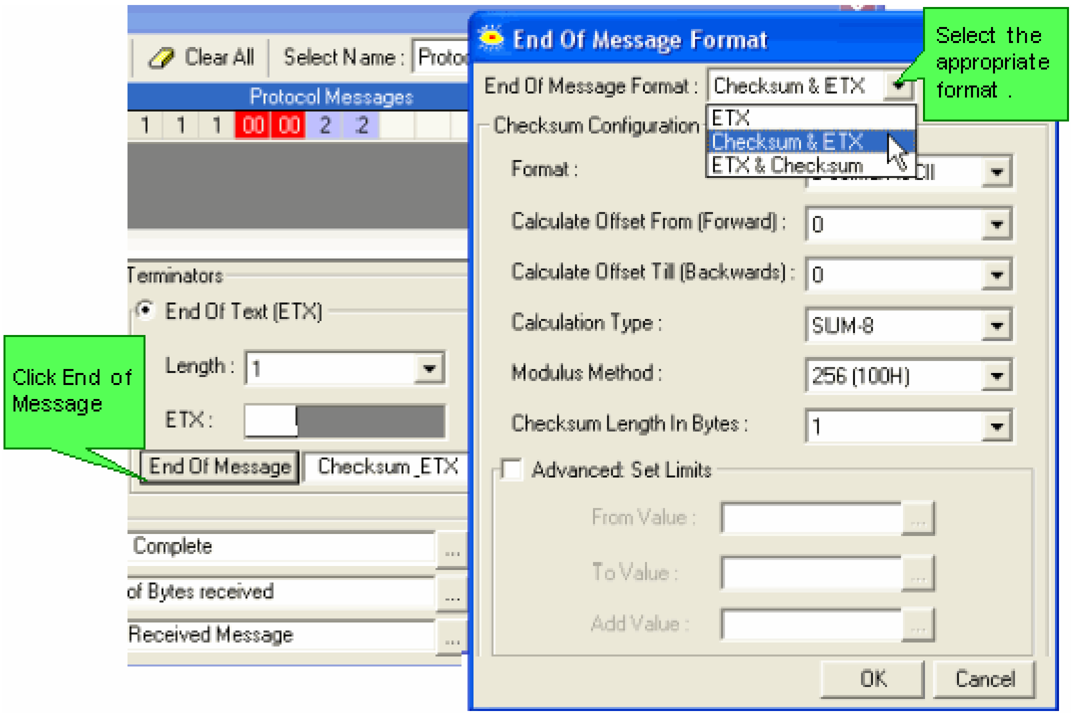
Start of Text (STX)	Check Start Of Text to use an STX parameter to indicate the beginning of a message. Length: Enter the number of bytes required by the external device's protocol to mark STX. STX Character: **Click a 'byte' box and select an ASCII character**. 
End of Text (ETX)	The ETX parameter indicates where the message ends. Length: Enter the number of bytes required by the external device's protocol to mark ETX. ETX: **Click a 'byte' box and Select an ASCII character**.

Dec	Hex	Description
0	00	(NUL)
1	01	(SOH)
2	02	(STX)
3	03	(ETX)



Dec	Hex	Description
0	00	(NUL)
1	01	(SOH)
2	02	(STX)

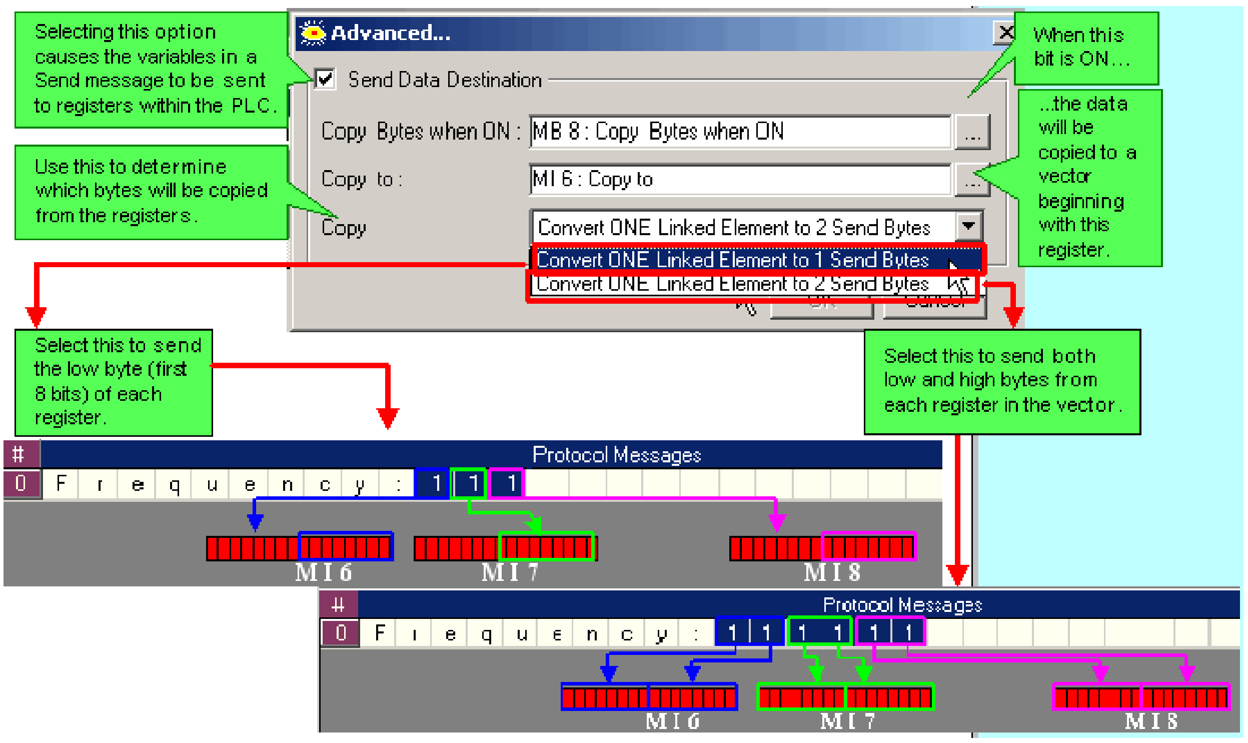
End of Message format:
- Note that the protocol may require that a checksum be either part of the message body or part of the End Of Message
- If your protocol requires a checksum as part of the End of Message,click the End of Message button to define **checksum placement and format.**
- Use the options to configure the checksum according to the requirements of your particular protocol.



End Of Message Format : Checksum & ETX
Checksum Configuration
Format :
Calculate Offset From (Forward) : 0
Calculate Offset Till (Backwards) : 0
Calculation Type : SUM-8
Modulus Method : 256 (100H)
Checksum Length In Bytes : 1
Advanced: Set Limits
From Value :
To Value :
Add Value :

Send Message Number	Determines which message in the FB will be sent. In the program, store a message index numbers in the linked register. When the FB is called, message whose index number currently in the linked MI will be sent.

Advanced Button


Selecting this option causes the variables in a Send message to be sent to registers within the PLC.
Use this to determine which bytes will be copied from the registers.
Select this to send the low byte (first 8 bits) of each register.
When this bit is ON...
...the data will be copied to a vector beginning with this register.
Select this to send both low and high bytes from each register in the vector.
Advanced...
Send Data Destination
Copy Bytes when ON : MB 8 : Copy Bytes when ON
Copy to : MI 6 : Copy to
Copy Convert ONE Linked Element to 2 Send Bytes
Convert ONE Linked Element to 1 Send Bytes
Convert ONE Linked Element to 2 Send Bytes
Protocol Messages
F r e q u e n c y : 1 1 1
MI 6
MI 7
MI 8
Protocol Messages
F r e q u e n c y : 1 1 1 1 1 1
MI 6
MI 7
MI 8

Protocol Messages

Messages can include:

- **Fixed Text**

Fixed text must be included if there is more than one message in a Scan. This enables the PLC to identify the message.

- **Control Characters**

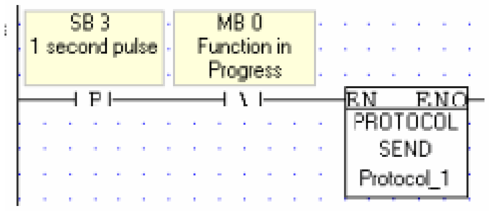
These can be added according to protocol requirements.

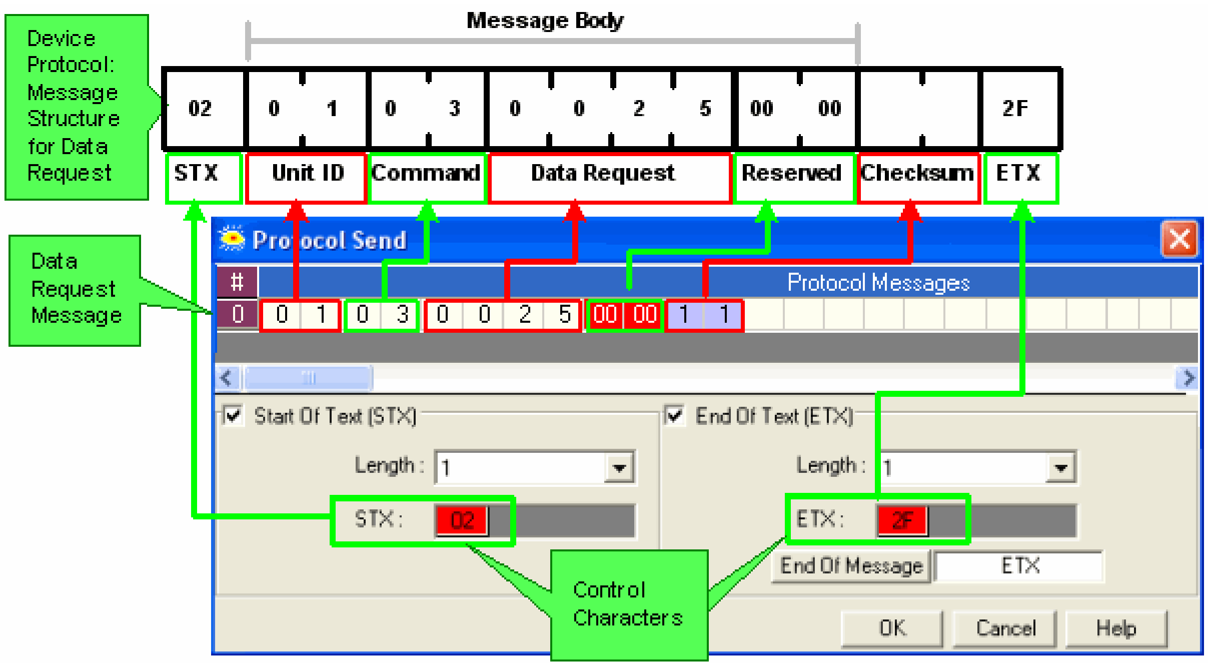
- **Variables**

These variables may be sent by the PLC: Binary, Numeric, Stream, List of Texts, Checksum, Date, and Time.

Example

Below, the Vision sends a data request to Unit 01. The request is Command 03, Read Register, and the register is Register 25.





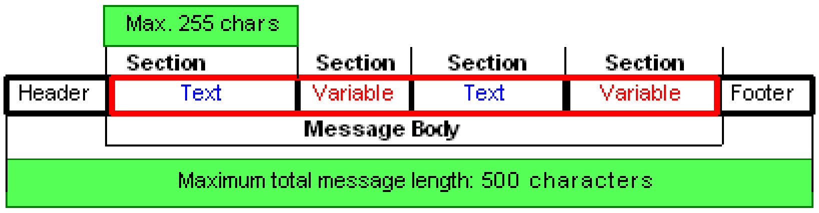
Device Protocol: Message Structure for Data Request
Message Body
02
0 1
0 3
0 0 2 5
00 00
2F
STX
Unit ID
Command
Data Request
Reserved
Checksum
ETX
Data Request Message
Protocol Send
#
Protocol Messages
0
0 1 0 3 0 0 2 5 00 00 1 1
Start Of Text (STX)
Length : 1
STX : 02
End Of Text (ETX)
Length : 1
ETX : 2F
End Of Message ETX
Control Characters
OK
Cancel
Help

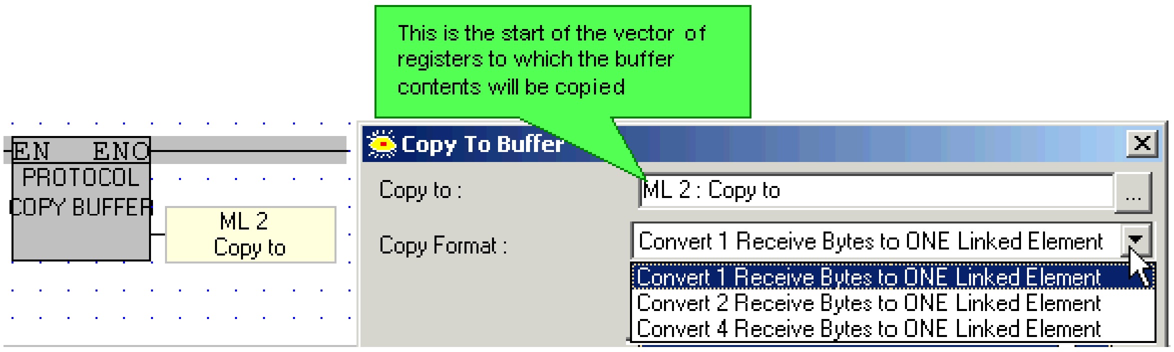
Protocol: Copy Buffer Contents

You can copy the current contents of the buffer which receives incoming messages at any point in your application by including the Copy to Buffer operation, which is located on the FBs menu.

This operation copies whatever information is in the buffer at the time the operation is called.

Note •	Note that copying the buffer contents is optional.
•	Maximum message length is 500 characters; a message may include up to 50 variables. Note that with in the body of a message, no section--whether text or variable--may exceed 255 characters.




This is the start of the vector of registers to which the buffer contents will be copied
EN ENO
PROTOCOL
COPY BUFFER
ML 2
Copy to
Copy To Buffer

Copy to : ML 2 : Copy to

Copy Format : Convert 1 Receive Bytes to ONE Linked Element

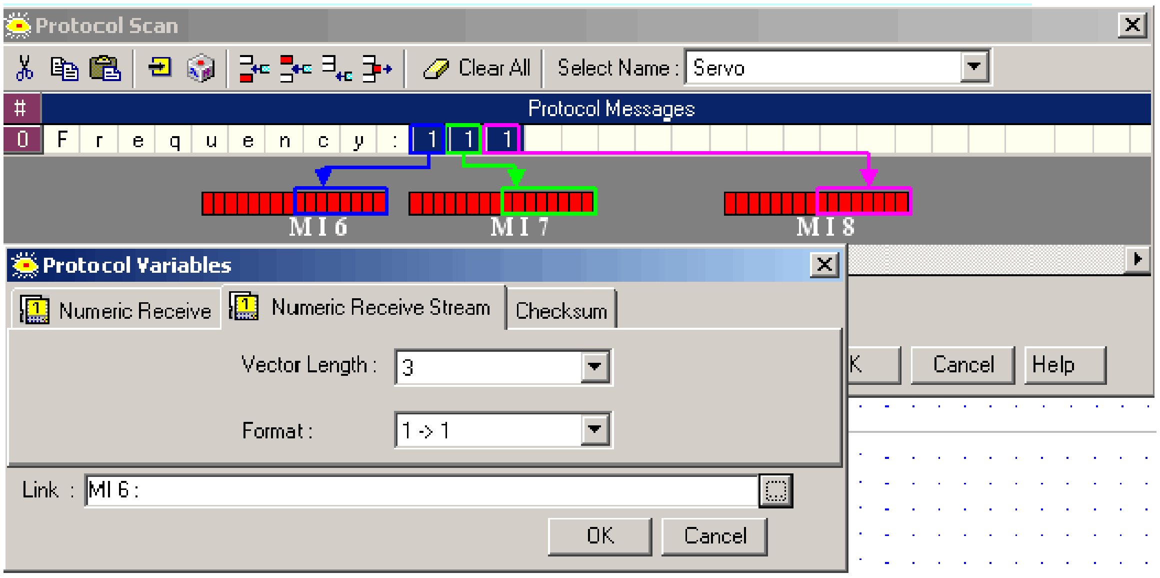
Convert 1 Receive Bytes to ONE Linked Element
Convert 2 Receive Bytes to ONE Linked Element
Convert 4 Receive Bytes to ONE Linked Element

Copy Format

The format determines how the received bytes are copied.

- **1->1**

Select this to copy the low byte (first 8 bits) of each received register value currently in the buffer to a vector defined in the PLC.



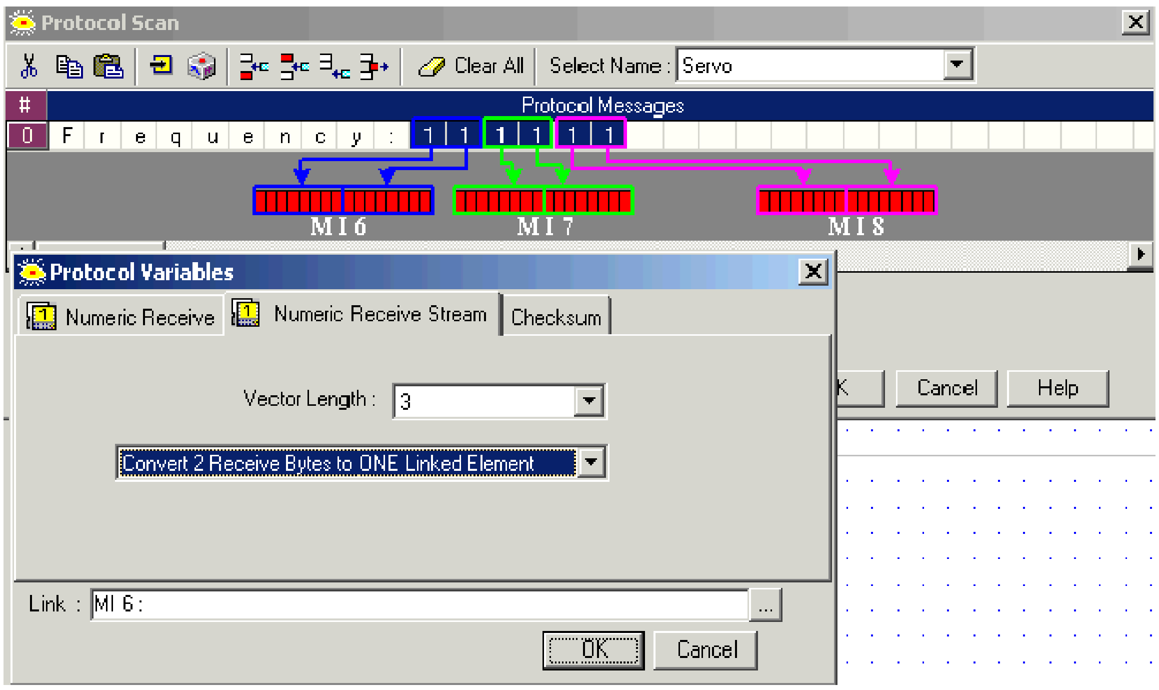
Vector Length : 3

Format : 1 -> 1

Link : MI 6 :

- **2->1**

Select this to copy both low and high bytes of each received register value currently in the buffer to a vector defined in the PLC. Note that this is an entire 16-bit register, and the first 2 bytes of a 32-bit register.



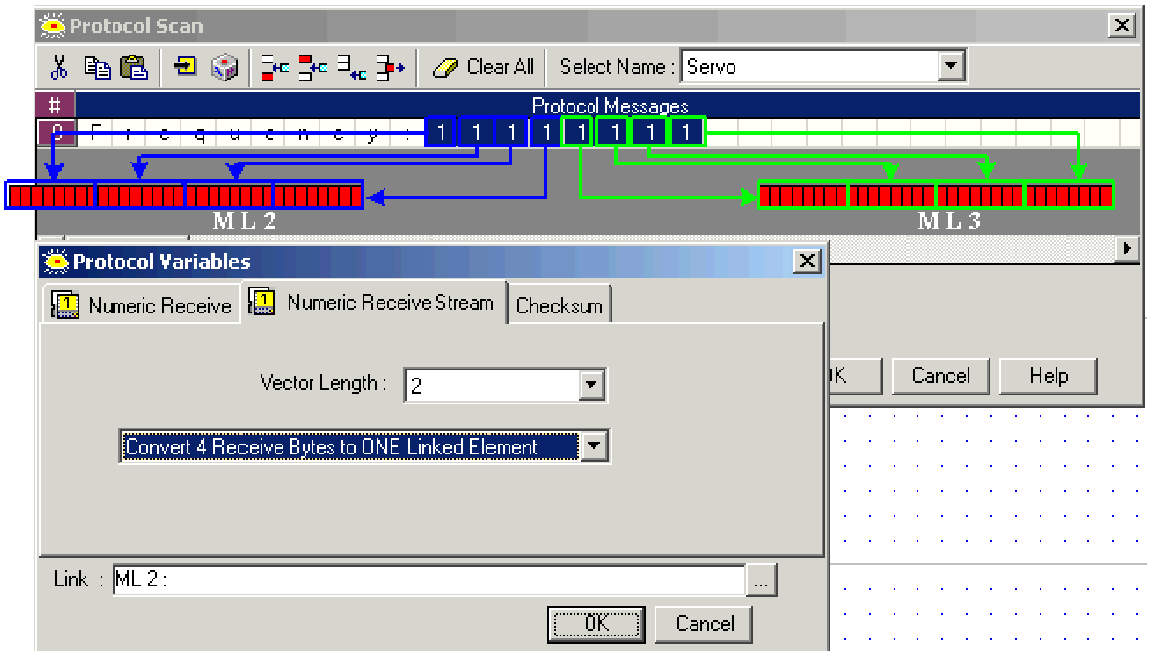
Vector Length : 3

Convert 2 Receive Bytes to ONE Linked Element

Link : MI 6 :

- **4->1**

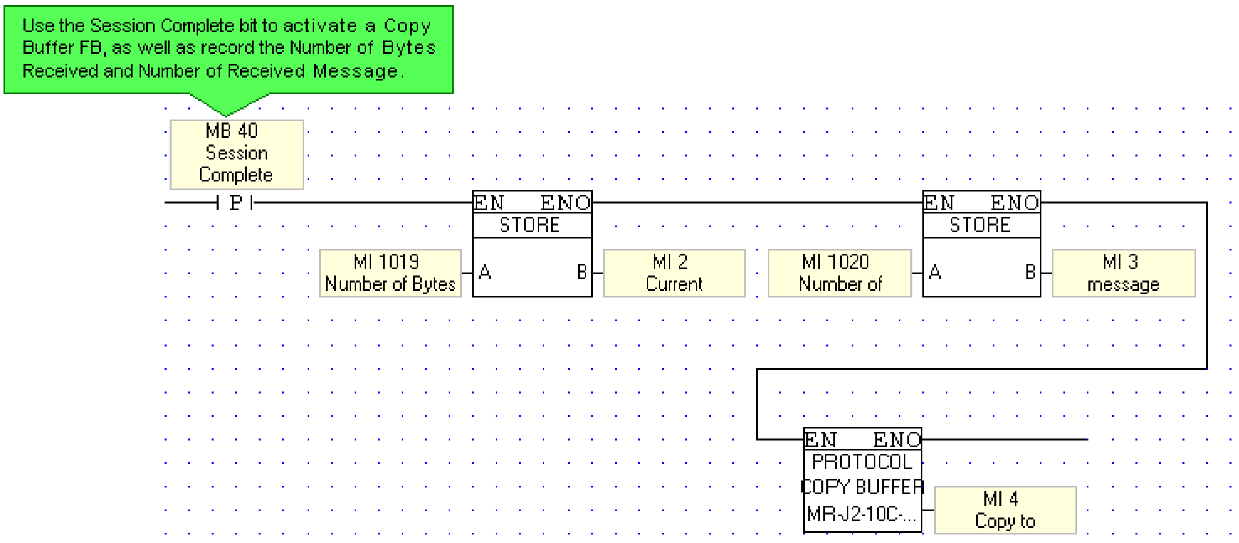
Select this to copy all 4 bytes of each 32-bit register currently in the buffer into a defined vector.



Vector Length : 2

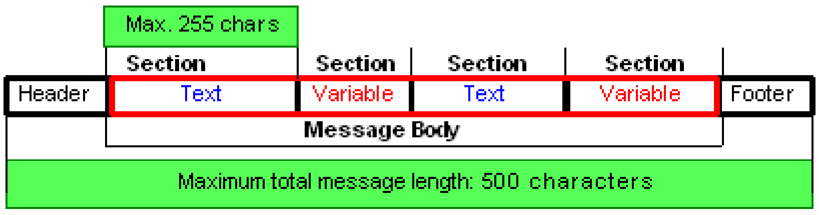
Convert 4 Receive Bytes to ONE Linked Element

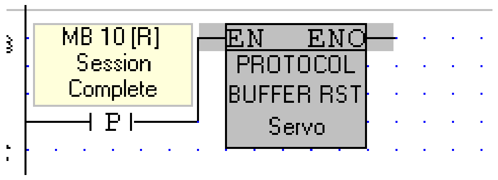
Link : ML 2 :


Use the Session Complete bit to activate a Copy Buffer FB, as well as record the Number of Bytes Received and Number of Received Message.

Protocol: Reset Buffer

In order to receive new messages, you must clear the communication buffer by using the Reset Buffer operation, which is located on the FBs menu.

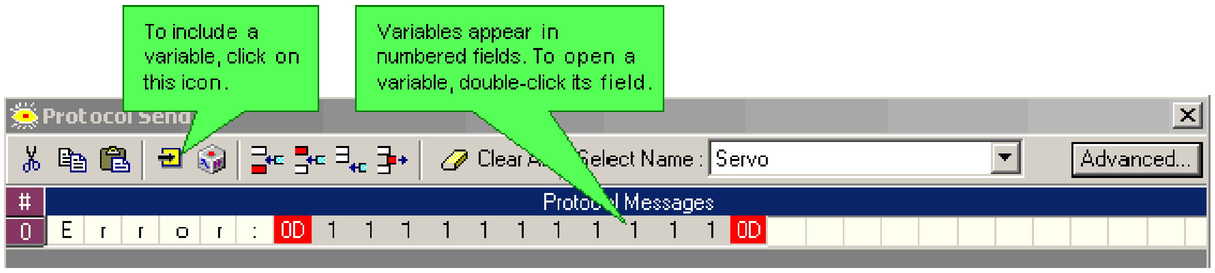
Note •	Note that the Session Complete bit turns OFF automatically when the Reset Buffer operation runs.
•	Maximum message length is 500 characters; a message may include up to 50 variables. Note that with in the body of a message, no section--whether text or variable--may exceed 255 characters.  Max. 255 chars Section Section Section Section Header Text Variable Text Variable Footer Message Body Maximum total message length: 500 characters



Variables

Variables enable you to send and receive a variety of dynamic data from your process.

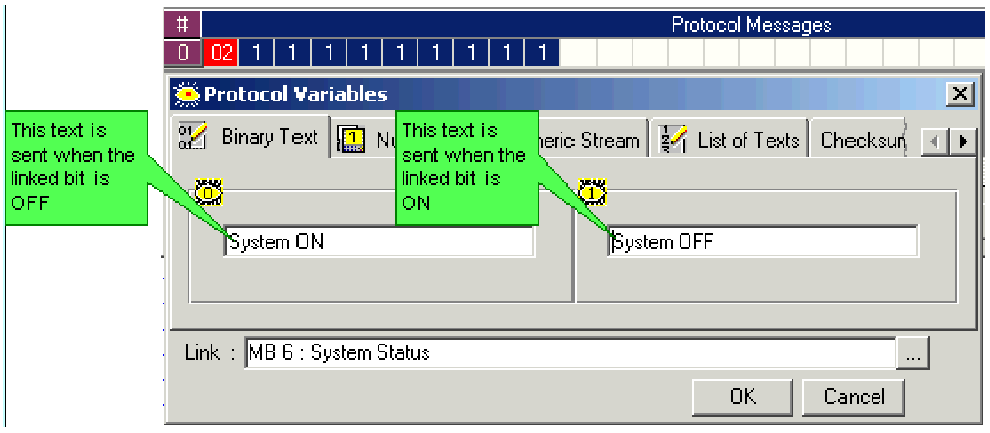
Variable Types: Send


To include a variable, click on this icon.
Variables appear in numbered fields. To open a variable, double-click its field.
Protocol Send
Clear A
Select Name : Servo
Advanced...
Protocol Messages
0 E r r o r : 0D 1 1 1 1 1 1 1 1 1 1 1 0D

The PLC sends messages via the Configuration linked to the Send operation. These are the types of variables that can be attached to an outgoing message.

- **Binary text**

Use this type of variable to send text based on the status of a linked bit.



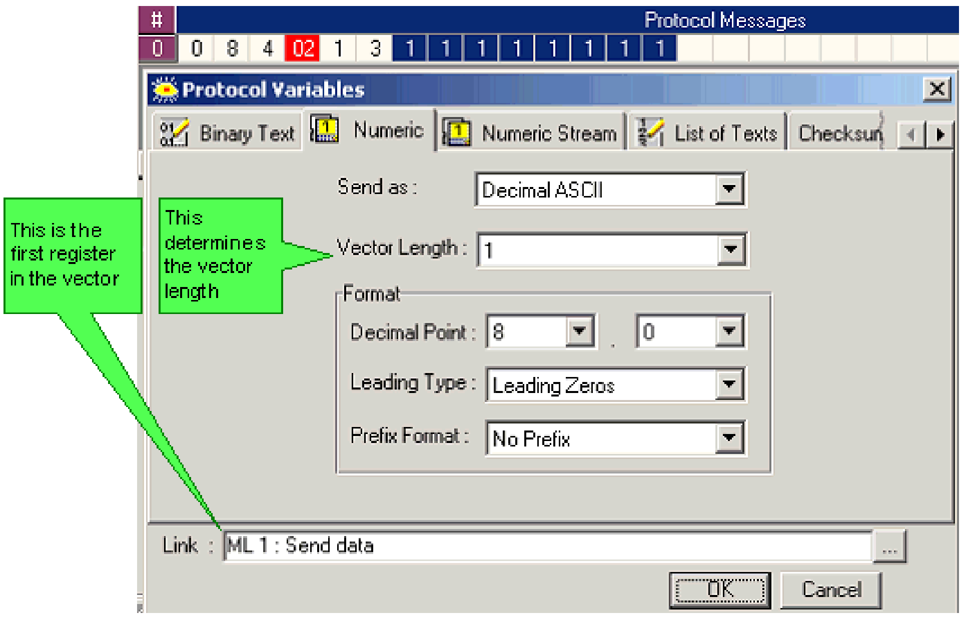
- **Numeric**

Use a Numeric variable to send a vector of registers. You can send register values as Binary, Hex as ASCII, or Decimal ASCII. The variable field automatically adjusts to fit the size of the data to be sent.

Note • Sending Hex as ASCII: the PLC transfers the value to ASCII, and sends it byte by byte, beginning from the most significant value.

Thus, the value 8F03BCA1 sent "Hex as ASCII" will be converted to Hex and sent as follows:

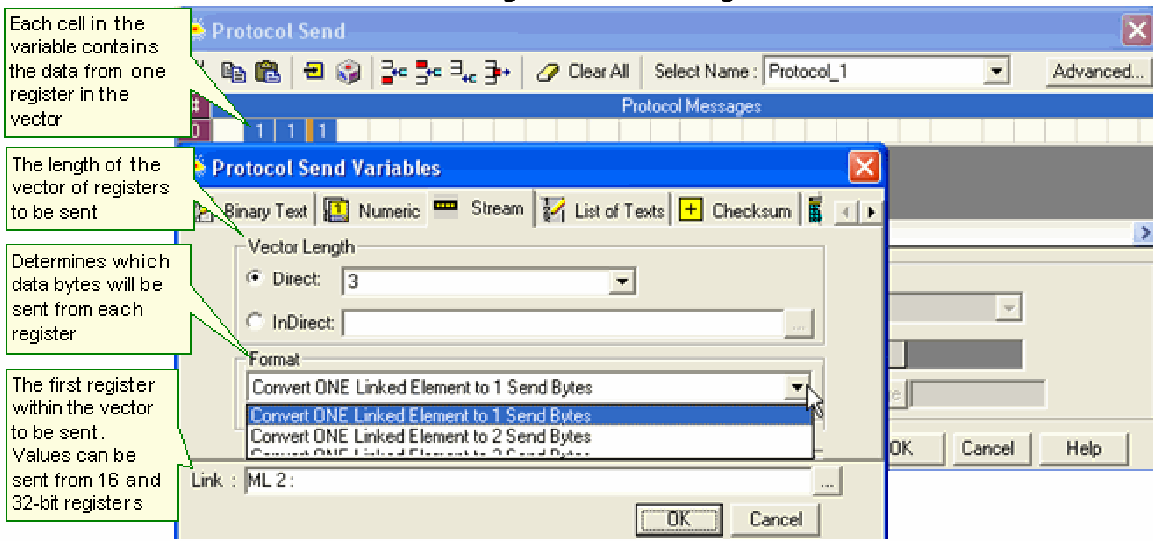
38, 46, 30, 33, 42, 43, 41, 31.



Send as : Decimal ASCII
Vector Length : 1
Format
Decimal Point : 8 . 0
Leading Type : Leading Zeros
Prefix Format : No Prefix
Link : ML 1 : Send data

- **Stream**

Use Stream to send a vector of registers to a target device.

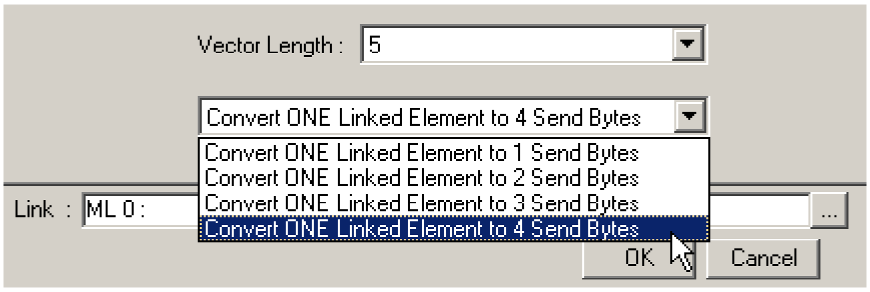

Each cell in the variable contains the data from one register in the vector
The length of the vector of registers to be sent
Determines which data bytes will be sent from each register
The first register within the vector to be sent. Values can be sent from 16 and 32-bit registers
Protocol Send
Clear All
Advanced...
Protocol Messages
Protocol Send Variables
Binary Text
Numeric
Stream
List of Texts
Checksum
OK
Cancel
Help

Select Name : Protocol_1
Vector Length
Direct: 3
InDirect:
Format
Convert ONE Linked Element to 1 Send Bytes
Convert ONE Linked Element to 1 Send Bytes
Convert ONE Linked Element to 2 Send Bytes
Link : ML 2 :

Note •	Values are sent in binary format, beginning from the LSB. Thus, the value 8F03BCA1 is sent as follows: A1, BC, 3, 8F.
•	In order to send floating values, select MF and the Convert 4 bytes into 1 linked element option. Note that the PLC uses the big-endian system; meaning that the most significant value in the sequence is stored at the lowest (first)storage address).

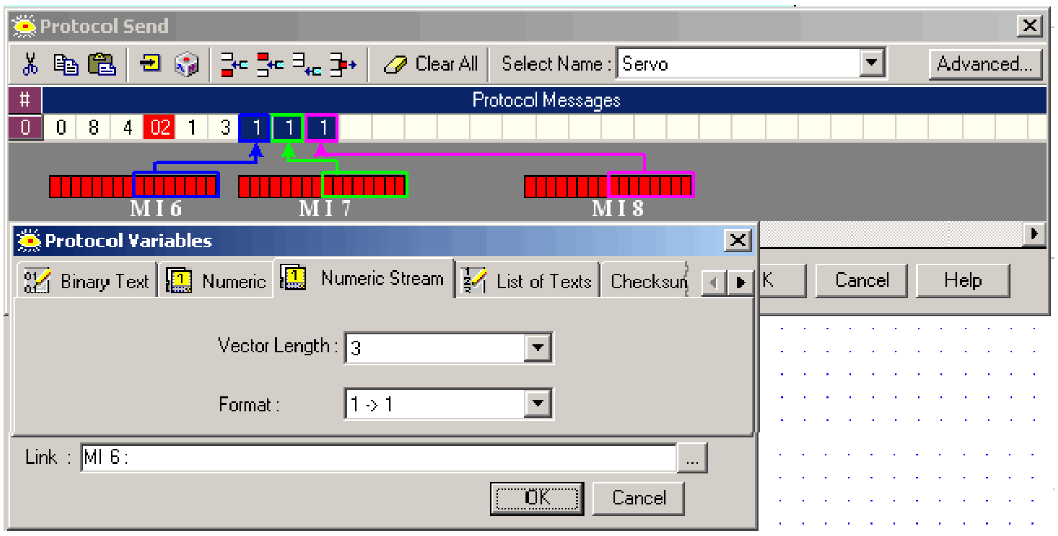
Stream Variable Format

The format determines which bytes are sent from within a register.


Vector Length : 5
Convert ONE Linked Element to 4 Send Bytes
Convert ONE Linked Element to 1 Send Bytes
Convert ONE Linked Element to 2 Send Bytes
Convert ONE Linked Element to 3 Send Bytes
Convert ONE Linked Element to 4 Send Bytes
Link : ML 0 :
OK
Cancel

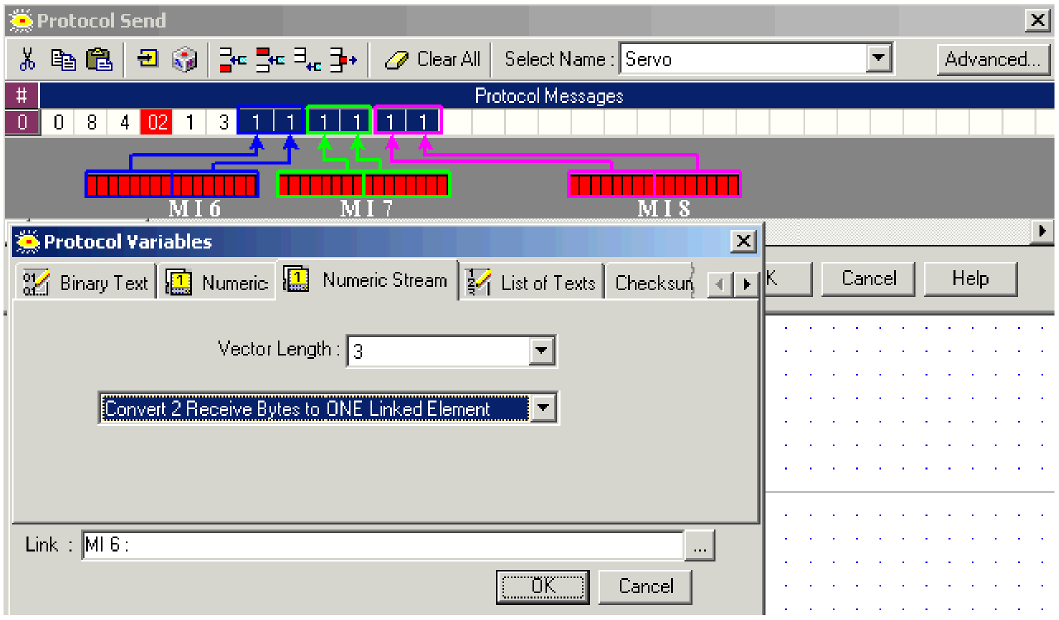
- **1->1**

Select this to send the low byte (first 8 bits) of each register in the vector.



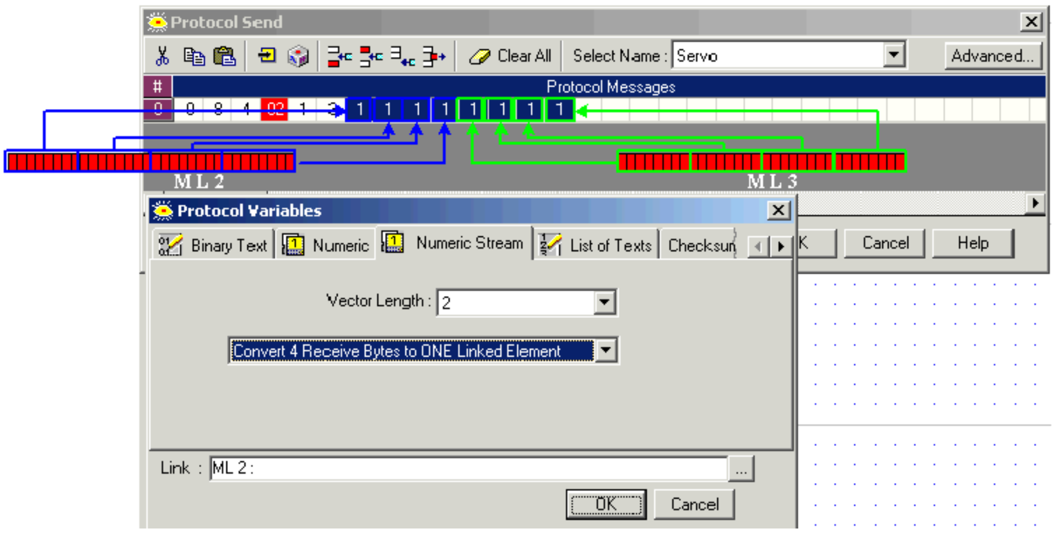
- **2->1**

Select this to send both low and high bytes from each register in the vector. Note that this is an entire 16-bit register, and the first 2 bytes of a 32-bit register.



- **4->1**

Select this to send all 4 bytes of each 32-bit register in a vector.



Vector Length : 2

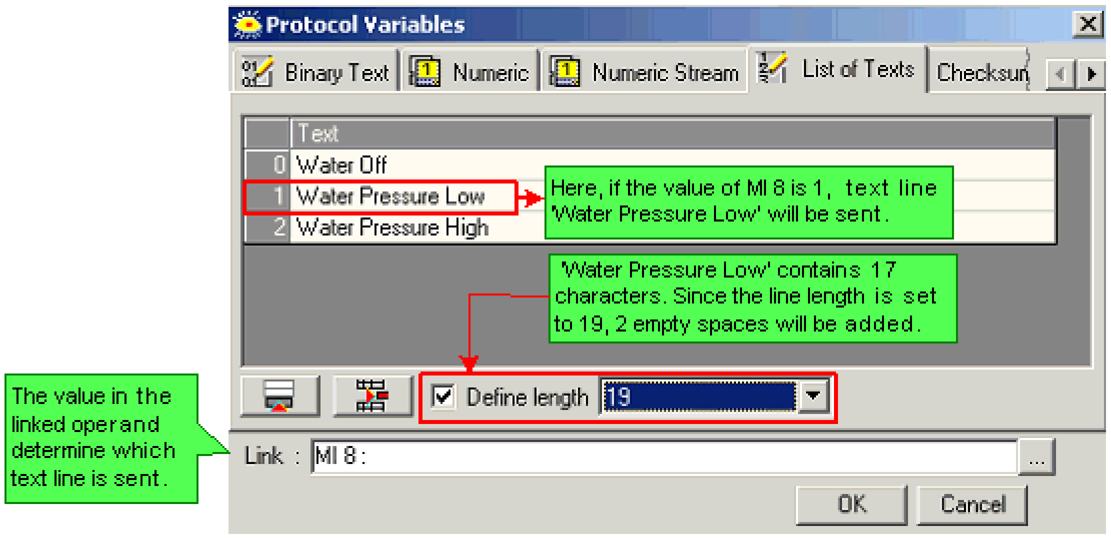
Convert 4 Receive Bytes to ONE Linked Element

Link : ML 2 :

- **List of texts**

This type of variable contains a list of numbered lines of text. The value within the linked operand 'points' to the number of a line within the list. When the operand value is equal to a particular line number, the text of that line is included in the variable.

Selecting the Define Length option enables you to make all of the text lines a uniform length, regardless of the number of characters a specific line may have. If a text line is shorter than the defined length, empty spaces will be added; if the text line is longer than the defined length, excess characters will be cut off.



	Text
0	Water Off
1	Water Pressure Low
2	Water Pressure High

Here, if the value of MI 8 is 1, text line 'Water Pressure Low' will be sent.

'Water Pressure Low' contains 17 characters. Since the line length is set to 19, 2 empty spaces will be added.

☑ Define length 19

The value in the linked operand determine which text line is sent.

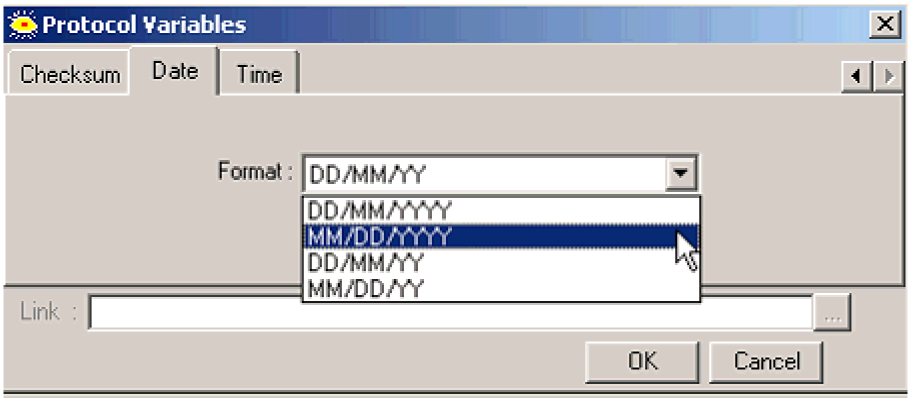
Link : MI 8 :

- **Checksum**

Via this variable, you can calculate a checksum and attach it to a message. Use the options to configure the checksum according to the requirements of your particular protocol.

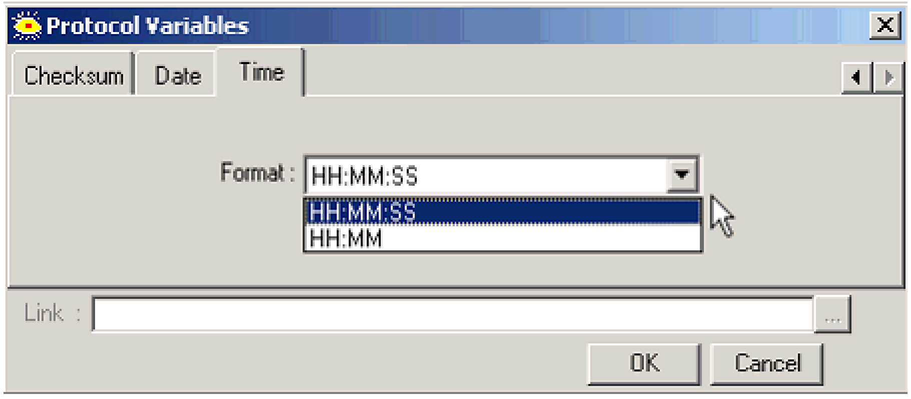
- **Date**

Use this variable to send the current date according to a selected format.



- **Time**

Use this variable to send the current time according to a selected format.



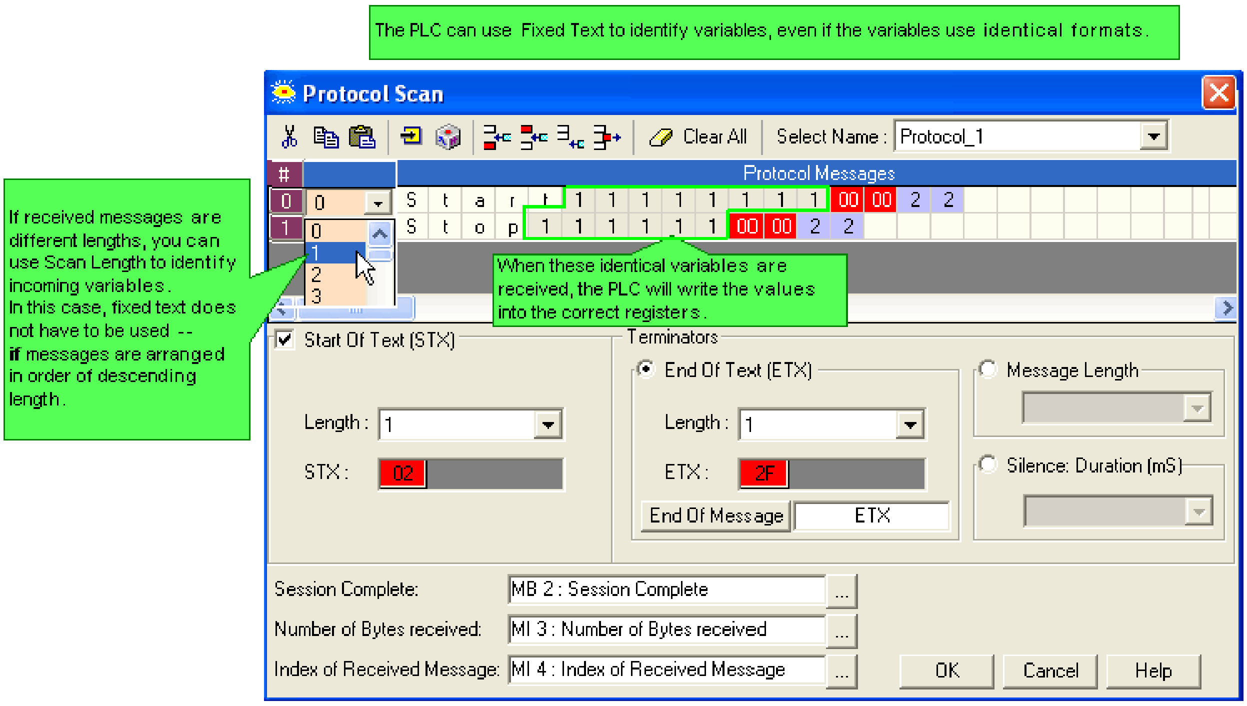
Variable Types: Scan

The PLC scans for messages that enter the system via the Configuration linked to the Scan operation. These are the types of variables that can be attached to an incoming message.

To correctly identify incoming variables, use either Scan Length, or include fixed text.

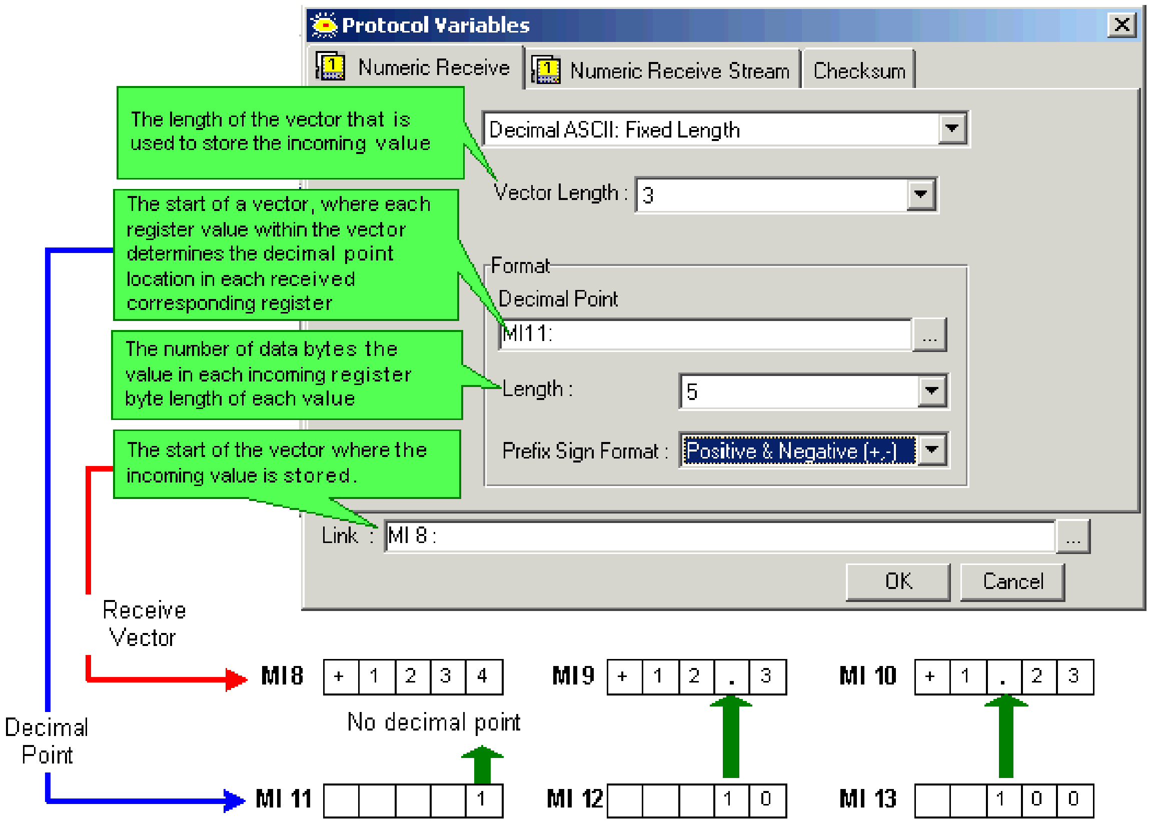
Notes •	Scan Length can be defined for each message via the drop-down box that follows Message index #. Scan length causes the Scan FB to use the length of a message to define whether an incoming message is legal. This parameter overrides Start Of Text and Terminator settings. If you use Scan Length, arrange the messages in ascending order, for example, message #0 may contain 3 bytes, #1 may contain 4 bytes, #2 may contain 6 bytes, etc.
•	If a Scan operation contains more than one message with identical variables that use the same format, the PLC will overwrite the variable values as they are received.

To enable the PLC to differentiate between the messages and their attached variables, include fixed text and/or control characters within the message.


The PLC can use Fixed Text to identify variables, even if the variables use identical formats.
If received messages are different lengths, you can use Scan Length to identify incoming variables. In this case, fixed text does not have to be used -- if messages are arranged in order of descending length.
When these identical variables are received, the PLC will write the values into the correct registers.
Protocol Scan
Clear All
Select Name : Protocol_1
Protocol Messages
Start Of Text (STX)
Length : 1
STX : 02
Terminators
End Of Text (ETX)
Length : 1
ETX : 2F
End Of Message ETX
Message Length
Silence: Duration (mS)
Session Complete: MB 2 : Session Complete
Number of Bytes received: MI 3 : Number of Bytes received
Index of Received Message: MI 4 : Index of Received Message
OK
Cancel
Help

- **Numeric Receive**

Use a Numeric variable to receive a vector of registers. You can receive register values as Binary, Hex as ASCII, or Decimal ASCII.

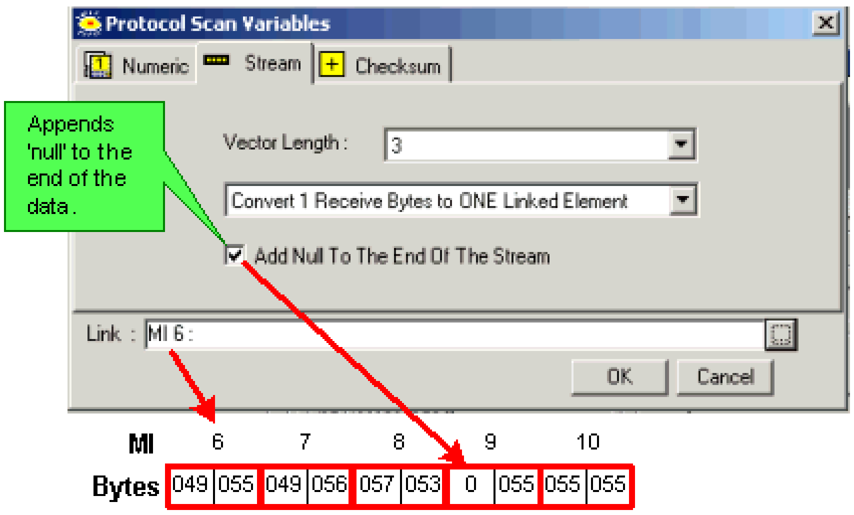

Protocol Variables
Numeric Receive
Numeric Receive Stream
Checksum
The length of the vector that is used to store the incoming value
The start of a vector, where each register value within the vector determines the decimal point location in each received corresponding register
The number of data bytes the value in each incoming register byte length of each value
The start of the vector where the incoming value is stored.
Decimal ASCII: Fixed Length
Vector Length : 3
Format
Decimal Point
MI 11 :
Length : 5
Prefix Sign Format : Positive & Negative (+,-)
Link : MI 8 :
OK
Cancel
Receive Vector
Decimal Point
MI 8 + 1 2 3 4
No decimal point
MI 9 + 1 2 . 3
MI 10 + 1 . 2 3
MI 11 1
MI 12 1 0
MI 13 1 0 0

- **Receive Stream**

Use Stream to receive a vector of register values from an external device and copy them into a defined vector of registers within the PLC.

Note • A vector is read either until the end of the defined vector length, or until a null character is encountered. By adding a null character to the end of the stream, you can mark the end of a data string. This can prevent other data, that might be present in a vector, from being added to the data string when the vector is read.

In order to send floating values, select MF and the Convert 4 bytes into 1 linked element option. Note that the PLC uses the big-endian system; meaning that the most significant value in the sequence is stored at the lowest (first)storage address).


Protocol Scan Variables
Numeric
Stream
Checksum
Appends 'null' to the end of the data.
OK
Cancel
MI 6 7 8 9 10
Bytes 049 055 049 056 057 053 0 055 055 055

Vector Length : 3

Convert 1 Receive Bytes to ONE Linked Element

[✓] Add Null To The End Of The Stream

Link : MI 6 :

- <u>**Checksum**</u>

Via this variable, you can process the checksum of an incoming message. Use the options to configure the checksum according to the requirements of your particular protocol.

Checksum Configuration

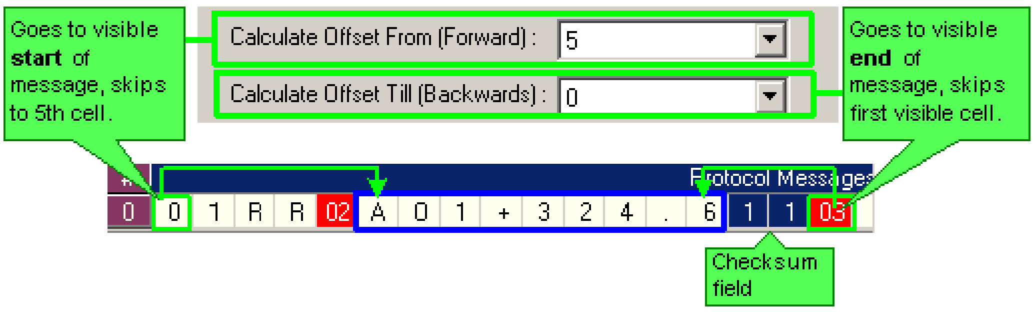
Use the options to configure the checksum according to the requirements of your particular protocol.

Format

Select the data representation method: Decimal ASCII, Hex as ASCII, or Binary.

Calculate Offset

The two Offset parameters determine the **delimiters of the data to be calculated for checksum.**



Calculate Offset From (Forward) : 5

Calculate Offset Till (Backwards) : 0

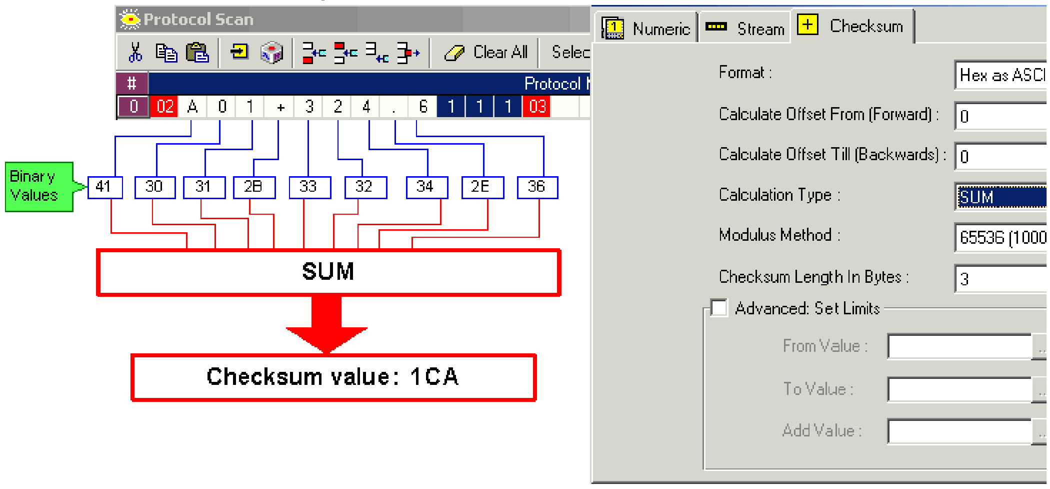
Goes to visible **start** of message, skips to 5th cell.
Goes to visible **end** of message, skips first visible cell.
Protocol Messages
0 | 0 | 1 | R | R | 02 | A | 0 | 1 | + | 3 | 2 | 4 | . | 6 | 1 | 1 | 03
Checksum field

Calculation Type

This is the type of calculation that will be performed on the data defined above.

SUM

Adds all of the data bytes in the selected cells into one sum.



Format : Hex as ASCI

Calculate Offset From (Forward) : 0

Calculate Offset Till (Backwards) : 0

Calculation Type : SUM

Modulus Method : 65536 (1000

Checksum Length In Bytes : 3

Advanced: Set Limits

From Value :

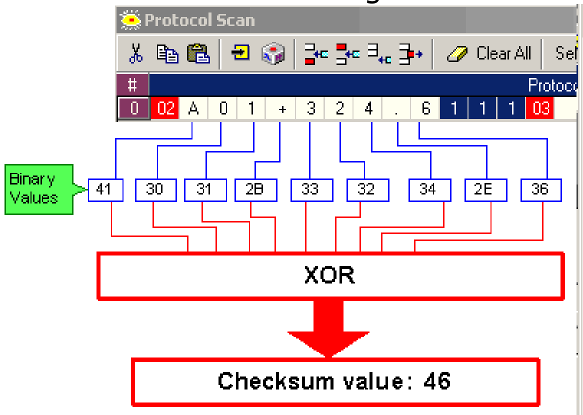
To Value :

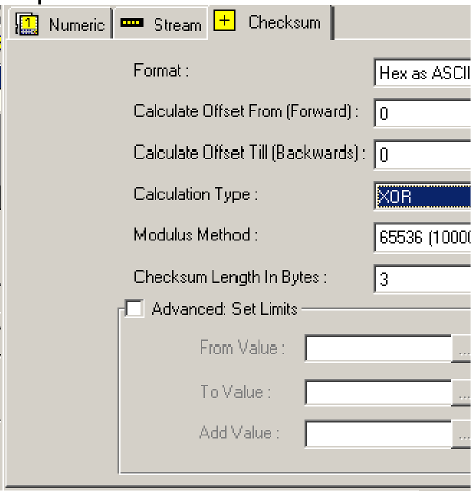
Add Value :

SUM	Checksum length / Format	1	2		3			4			
Modulo 256 (100H)	Binary (CA)	CA	0	CA	0	0	CA	0	0	0	CA
	Decimal (202)	32 2	30 0	32 2	32 2	30 0	32 2	30 0	32 2	30 2	32 2
	Hex as ASCII (CA)	41 'A'	43 'C'	41 'A'	30 0	43 'C'	41 'A'	30 0	30 0	43 'C'	41 'A'
Modulo 65536 (10000H)	Binary (1CA)	CA	1	CA	0	1	CA	0	0	1	CA
	Decimal (458)	38 8	35 5	38 8	34 4	35 5	38 8	30 0	34 4	35 5	38 8
	Hex as ASCII (1CA)	41 'A'	43 'C'	41 'A'	31 '1'	43 'C'	41 'A'	30 '0'	31 '1'	43 'C'	41 'A'

XOR

Calculate checksum using exclusive-OR operation.





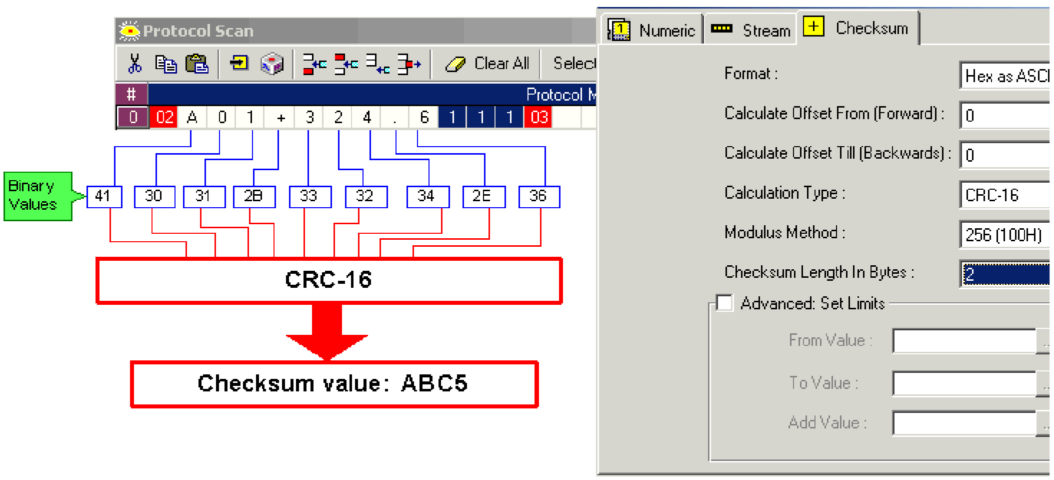
Format :	Hex as ASCII
Calculate Offset From (Forward) :	0
Calculate Offset Till (Backwards) :	0
Calculation Type :	XOR
Modulus Method :	65536 (1000
Checksum Length In Bytes :	3
Advanced: Set Limits	
From Value :	
To Value :	
Add Value :	

XOR	Checksum length / Format	1	2		3			4			
Modulo 256 (100H)	Binary (46)	46	0	46	0	0	46	0	0	0	46
	Decimal (70)	30 0	37 7	30 0	30 0	37 7	30 0	30 0	30 0	37 7	30 0
	Hex as ASCII (46)	36 '6'	34 '4'	36 '6'	30 '0'	34 '4'	36 '6'	30 '0'	30 '0'	34 '4'	30 '0'
Modulo 65536 (10000H)	Binary (46)	46	0	46	0	0	46	0	0	0	46
	Decimal (70)	30 0	37 7	30 0	30 0	37 7	30 0	30 0	30 0	37 7	30 0
	Hex as ASCII (46)	36 '6'	34 '4'	36 '6'	30 '0'	34 '4'	36 '6'	30 '0'	30 '0'	34 '4'	30 '0'

CRC: acronym for Cyclic Redundancy Check, a procedure used in checking for errors in data transmission. A complex polynomial is used to generate a number that is based on the transmitted data. The sending device performs the calculation before transmission, and then sends the result to the receiving device. The receiving device repeats the same calculation after it receives the data. If both devices arrive at the same result, the transmission is assumed to be error-free. This is called a redundancy check because each transmission includes not only data but extra (redundant) error-checking values

CRC-16

Calculates checksum according to CRC-16.



Format : Hex as ASCI
Calculate Offset From (Forward) : 0
Calculate Offset Till (Backwards) : 0
Calculation Type : CRC-16
Modulus Method : 256 (100H)
Checksum Length In Bytes : 2
Advanced: Set Limits
From Value :
To Value :
Add Value :

CRC-16	Checksum length / Format	1	2		3			4			
	Binary (ABC5)	C5	AB	C5	0	AB	C5	0	0	AB	C5
	Decimal (43973)	33 '3'	37 '7'	33 '3'	39 '9'	37 '7'	33 '3'	33 '3'	39 '9'	37 '7'	33 '3'
	Hex as ASCII (ABC5)	35 '5'	43 'C'	35 '5'	42 'B'	43 'C'	35 '5'	41 'A'	42 'B'	43 'C'	35 '5'

CRC-CCITT

This function computes a Cyclic Redundancy Code of the 8-bit character string, using X16 + X12 + X5 + 1 as the polynomial. The optional parameter seed may supply an initial value, which allows for running CRC calculations on multiple strings. If the parameter seed is not specified, a default value of 65,535 (216-1) is assumed.
Four types of CRC – CCITT are available:
CRC – CCITT 1: Checksum performed on 16-bit integers, initialized to zero value.
CRC – CCITT 2:: 16-bit integer, initialized to value 0xFFFF.
CRC – CCITT 3: Checksum performed on bytes, initialized to zero value.
CRC – CCITT 4: bytes, initialized to 0xFFFF.

C code

```
//-----------------------------------------------------------------------------
------------------------------
word wCalcCRC_CCITT_WordRes( byte * bpBuff, word wBuffLen,
word wInitVal )
{
byte bShifter;
byte bData;
word wCarry;
word wCRC;
word wIndex;
int iIndex2;
wCRC = wInitVal;
 for ( wIndex = 0; wIndex < wBuffLen; wIndex+=2 )
```

```
{
for ( iIndex2 = 1; iIndex2 > -1 ; iIndex2 --)
{
bShifter = 0x80;
bData = bpBuff[iIndex2];
do
{
wCarry = wCRC & 0x8000;
wCRC <<= 1;
if ( bData & bShifter )
{
wCRC++;
}
if ( wCarry)
{
wCRC ^= 0x1021;
}
bShifter >>= 1;
} while (bShifter);
}
 bpBuff+=2;
}
return wCRC;
}
//----------------------------------------------------------------------
-------------------
word wCalcCRC_CCITT_ByteRes( byte * bpBuff, word wBuffLen,
word wInitVal )
{
byte bShifter;
byte bData;
word wCarry;
word wCRC;
wCRC = wInitVal;
while( wBuffLen--)
{
bShifter = 0x80;
bData = *bpBuff;
 bpBuff++;
do
{
wCarry = wCRC & 0x8000;
wCRC <<= 1;
if ( bData & bShifter )
{
wCRC++;
}
if ( wCarry)
{
wCRC ^= 0x1021;
}
bShifter >>= 1;
} while (bShifter);
}
return wCRC;
}
```

2's complement

This is calculated as follows:

```
Checksum = 0
For Stardoms to Endorses do
Checksum = Checksum + Netherworlds
End For
Checksum = (Checksum  XOR 0xFFFFFFFF) + 1
```

Modulus method

256 (100H): 8 bits

Select to store the first 8 bits (low data byte) of the result into the checksum field.

65536 (10000H): 16 bits

Select to store the entire result into the checksum field.

Checksum Length

This determines the length of the field which will hold the calculated checksum result.

Example

If the result is **7563** and the 16-bit modulus method is selected, the results stored in checksum fields of different lengths as follows:

	Length: 1	Length: 2	Length: 3
Binary	63	7563	0075, 63
Decimal	33	36, 33	35, 36, 33
Hex as ASCII (7563 =1D8B)	42	38, 42	44, 38, 42

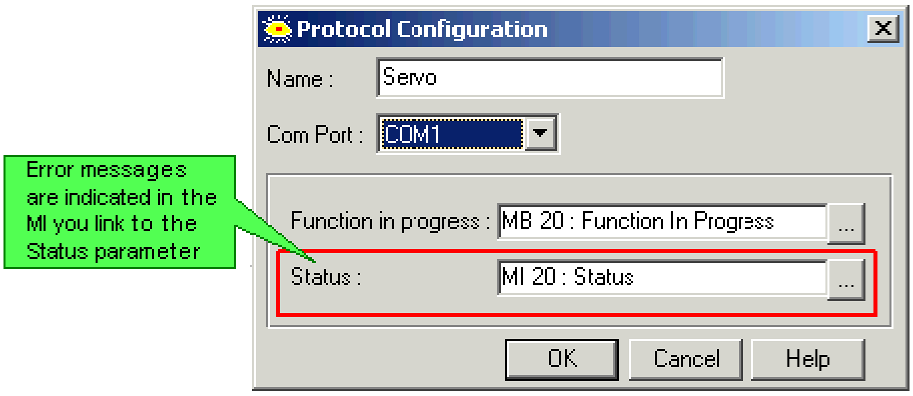
Advanced: Set Limits

From Value and To Value determine a range for the checksum result ;if the final value falls within these limits, the function can automatically add a set value to the calculated checksum.

Protocol Status Operands and Messages

The Status Operands show the state of Protocol communications.

All of the Status operands linked to Protocol FBs should be assigned Power-up Values; bits should be reset, and registers initialized to 0.


Error messages are indicated in the MI you link to the Status parameter

Protocol Configuration

Name : Servo

Com Port : COM1

Function in progress : MB 20 : Function In Progress

Status : MI 20 : Status

Protocol Operation Status Operands

When you place a Protocol Configuration your application, the linked operands indicate the status of Protocol operations.

Function in Progress Shows status of Protocol Configuration	MB	**Turns ON when:** Configuration begins When **Send** or **Scan** begins **Note** • Messages cannot be sent while this MB is ON. Use an indirect contact for this MB as an activating condition for Send operations.	**Turns OFF when** The Protocol: Configuration is finished. When Scan is complete, when the PLC receives the message's STX and terminator. The Status Message indication changes
Status Messages	MI	Automatically initialized to 0 when a **Protocol** operation is activated. Updated at the end of each attempt to communicate via the **Protocol** Configuration. Indicates status of **Protocol** communications, according to the table below. Note that the current value always shows the most **recent** status. Value Error Message Indication 0 No Error Operations successful 1 Send: General ErrorIndicates an unknown system error. 2 Send: COM Port busy The COM port that is selected in the active Protocol Configuration cannot currently be accessed. 3 Send: Invalid Message Index The message that is referenced in the Send operation does not exist. 4 Send: Invalid Operand Type The message references an operand type that does not exist in the system. 5 Send: Invalid Operand Address The message references an operand address that does not exist in the system. 6 Send: Data Overflow The message data exceeds 512 bytes. 7 Send: Format Error The message contains variable fields that are not long enough for the attached variable data. 257 Receive: General Error Indicates an unknown system error. 258 Receive: Invalid message The received message does not exist in the system. 259 Receive: Checksum Error The message may have been corrupted during transmission.	

Examples

Basic Example

How does the Protocol FB communicate data between Vision controllers and other devices?

A device, such as a magnetic card reader, may use its own, proprietary protocol. If you know the protocol's structure, you can use the Protocol FB to structure messages accordingly. This enables a Vision controller to exchange data with the device using the device's own protocol.

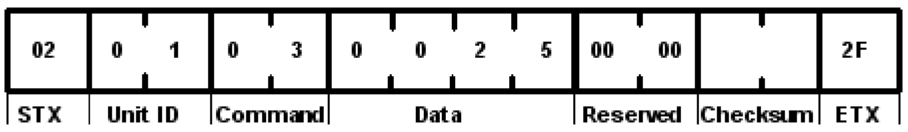
The example below shows how a Vision controller can, via COM port 2, read Register 25 within a device that has the ID number of 01. The device uses the simple protocol shown below.

Device Protocol

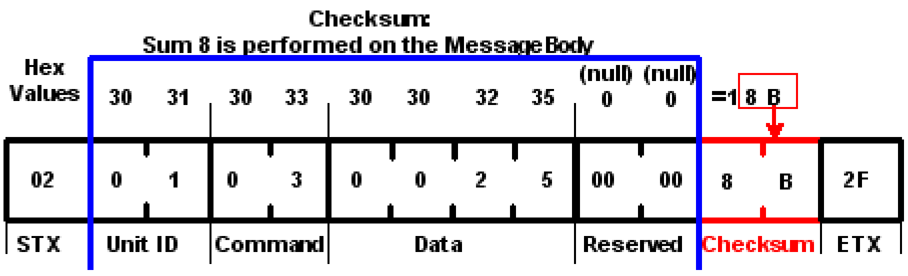
Parameter name	**Length (bytes)**	**Value**	**Comments**
STX	1	STX 02(hex)	All messages must begin with the STX character
Unit ID	2	ID# range= 0-99	All messages must contain the ID number of the device that the controller is communicating with
Command	2	03=Read Register 04=Read Bit 05=Write Register	Commands are embedded in Data Request messages.
Data (Request)	4		Vision to Device
Data (Response	8		Device to Vision
Reserved	2	2 Null characters=00,00	These characters must be included in all messages.
Checksum	2	Sum 8	
ETX	1	/ backslash) 2F(hex)	All messages must end with the '/'character

Message Structure: Data Request (Vision to device)

The device can only receive request messages that are structured as shown below.



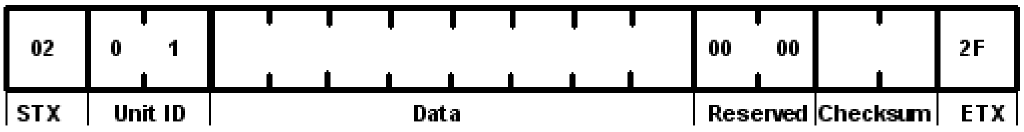
Checksum



Checksum:
Sum 8 is performed on the Message Body
Hex Values
30 31 30 33 30 30 32 35 (null) 0 (null) 0 =18 B
02 0 1 0 3 0 0 2 5 00 00 8 B 2F
STX
Unit ID
Command
Data
Reserved
Checksum
ETX

Message Structure: Data Response (device to Vision)

The device sends data in messages that are structured as shown below.



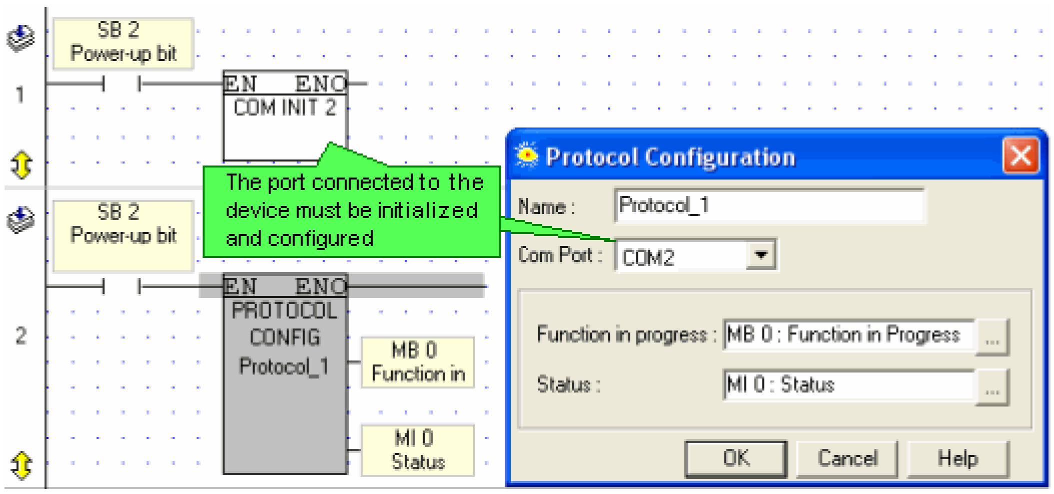
Protocol Ladder Requirements

In order to enable a Vision controller to request and receive data from a device, the controller's Ladder application must contain:

- A COM Init FB to initialize the serial port that will be linked to the device.
- A Protocol Configuration FB set to use the serial port.
- A Protocol Send FB, which enables the controller to send data request messages to the device.
- A Protocol Scan FB, which enables the controller to receive data response messages from the device. Note that in order to receive new messages, you **must clear the communication buffer after each message by using the Reset Buffer FB.**

COM Init and Configuration

Since COM Port 2 is connected to the device, the application initializes and configures COM 2 at Power-up.

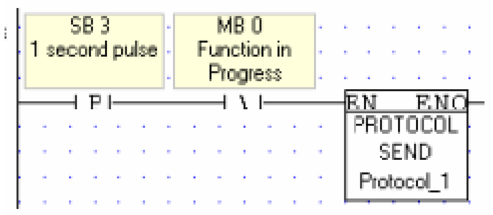


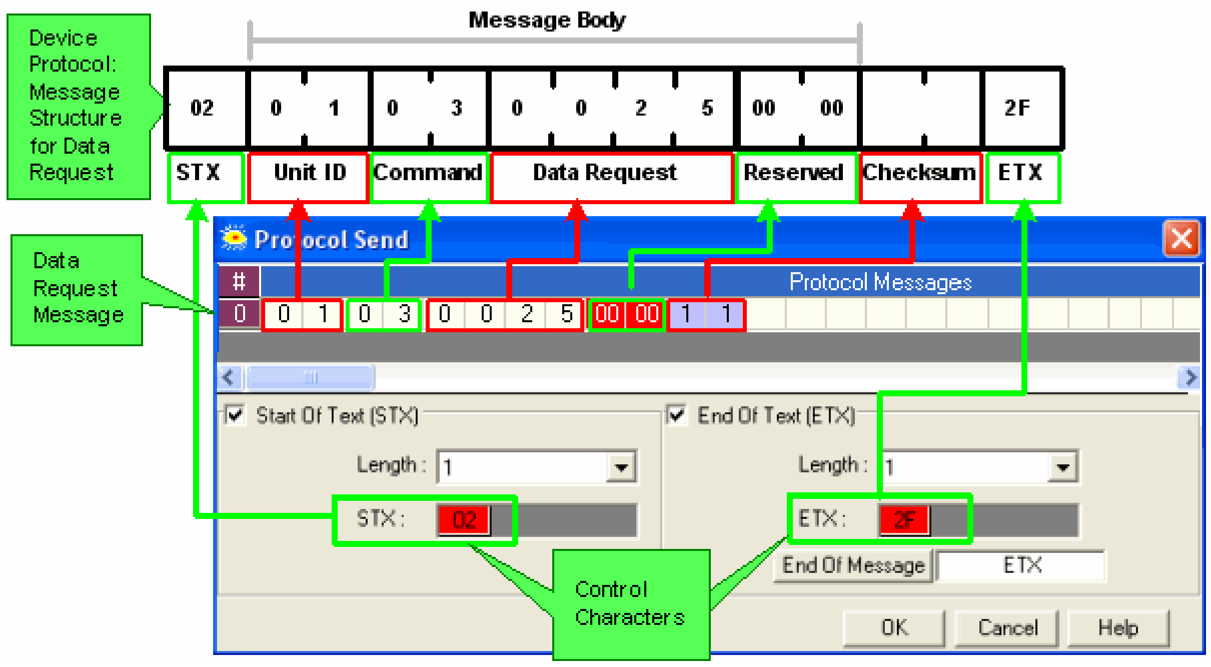
Name : Protocol_1
Com Port : COM2
Function in progress : MB 0 : Function in Progress
Status : MI 0 : Status

Send: Data Request

The application requests data via the Protocol Send FB. The Send FB enables you to structure messages that conform to any protocol's requirements.

Below, the Vision sends a data request to Unit 01. The request is Command 03, Read Register, and the register is Register 25.



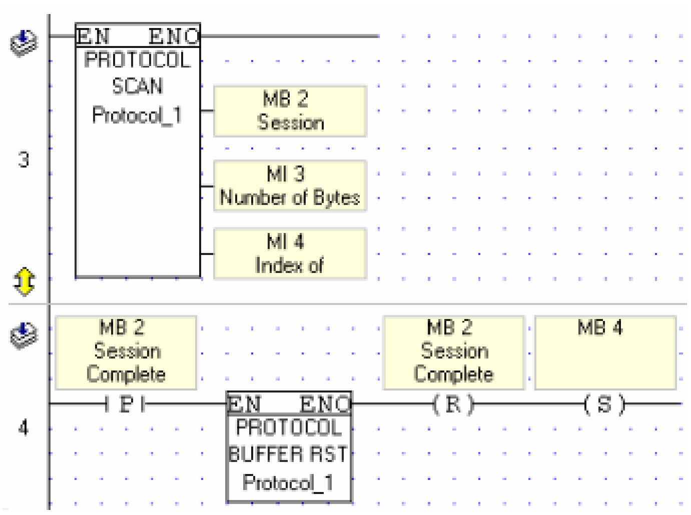


Scan: Receiving Messages

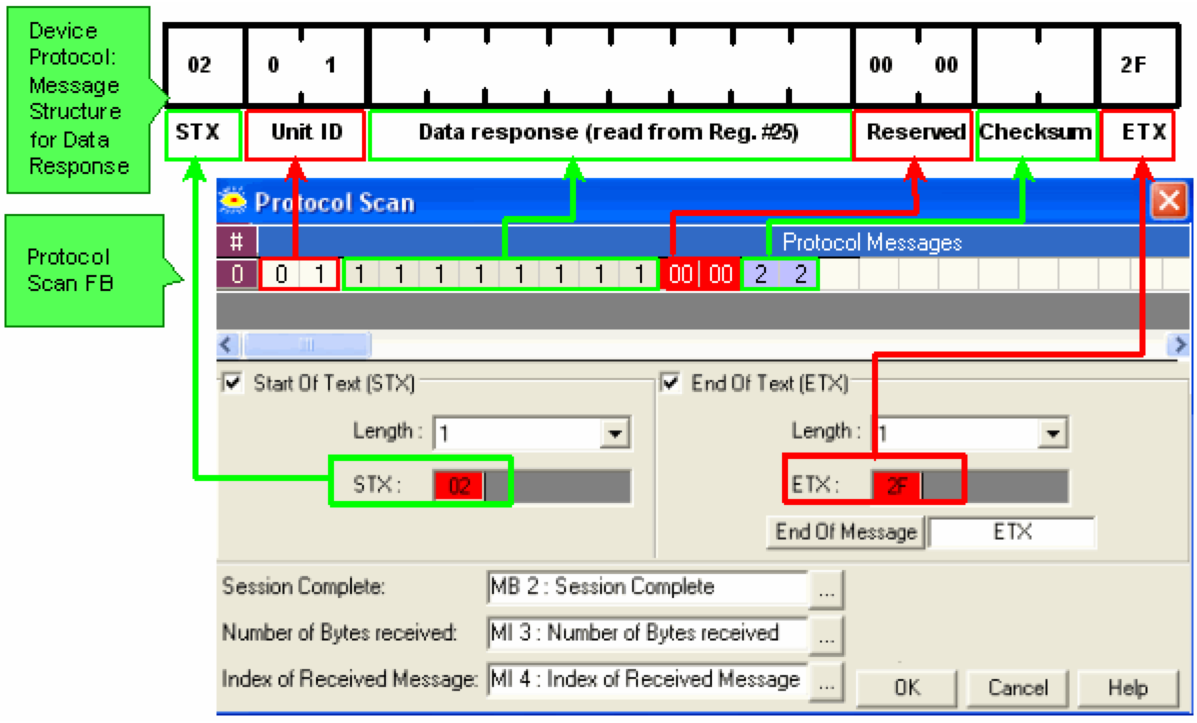
A Protocol Scan FB contains messages. Activating the Scan causes the Vision to check if it has received any of the messages contained within the Scan FB. Note that the controller can only receive messages that are defined in the Scan FB.

Note • In order to receive new messages, you **must** clear the communication buffer by using the Reset Buffer operation. You can save the contents of the buffer at ay time by using the Copy Buffer operation.

Below, the application enables the Vision to receive a response to Command 03. Note that Buffer reset follows the Scan.



Note that the Scan message contains a Data Response field that is long enough to contain the contents of a register.



Start Of Text (STX)
Length : 1
STX : 02

End Of Text (ETX)
Length : 1
ETX : 2F
End Of Message ETX

Session Complete: MB 2 : Session Complete
Number of Bytes received: MI 3 : Number of Bytes received
Index of Received Message: MI 4 : Index of Received Message

Advanced Mitsubishi Frequency Converter

The information below is intended to accompany the sample application Mitsubishi.vlp. This application shows how to exchange data between networked Mitsubishi frequency converters and a Vision controller.

Data exchange is performed using the Mitsubishi communication protocol, via VisiLogic's communication Protocol FB. By modifying the FB's parameters, you can exchange data between Vision controllers and external devices using many different protocols.

About the Mitsubishi Protocol

The Mitsubishi manual for the converter used in the example may be found by accessing the Mitsubishi site at:

http://www.meau.com/eprise/main/Web_Site_Pages/Public/Documents_Downloads/P-DD-Technical-Manuals,

entering SH(NA)3197-B under **Title**, and then pressing **Search**. Note that this manual is in .pdf format and requires you to have Adobe's Acrobat Reader installed on your system.

The manual includes complete protocol requirements. The elements used in the sample application to implement the Mitsubishi protocol are presented below.

Read Commands

Command	Data No.	Descriptions	Frame Length
01	80-80E	Data Value and processing information of status display	12
02	00	Current alarm number	4
	90	Absolute position of motor end in pulse unit	8
	91	Absolute position in command unit	8
05	00-35	Current value of corresponding parameters (decimal numbers in data number correspond to the parameter numbers)	8
33	10-15	Alarm number	4
	20-15	Alarm Time	8
35	80-80E	Data value and processing information of status display when alarm occurs	12
6D	Register number	Register amount (single register)	8
6E	Register number	Register amount (double register)	8

Write

Command	Data No.	Descriptions	Setting Range	Frame Length
81	00	Status Display clear	1EA5	4
82	00	Alarm reset	1EA5	4
84	00-35	Write to corresponding parameters (the decimal number in the data number corresponds to parameter number)	Areas according to parameter #	8
92	60	Communication device input signal		8
B8	00-3F	Program data		8
B9	Register number	Register amount (single register)		8
BA	Register number	Register amount (double register)		8

Station ID

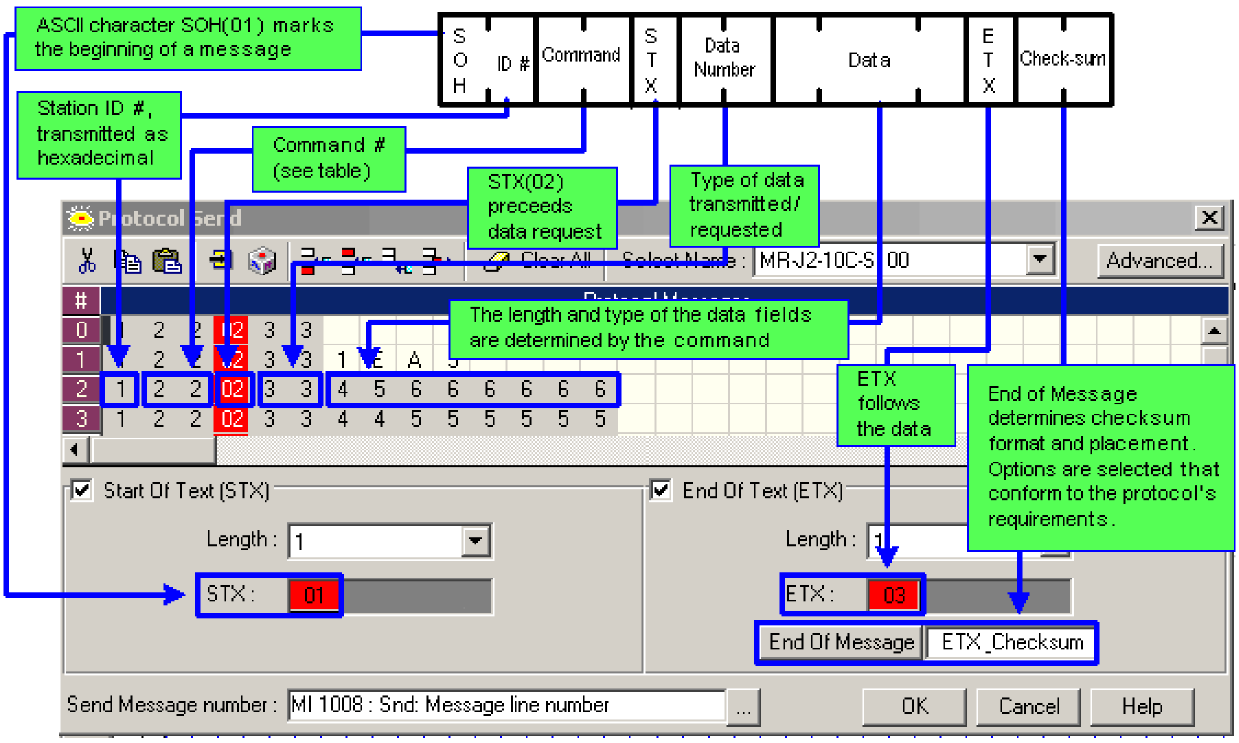
Each converter is assigned a station number. Station numbers are transmitted in hexadecimal.

ID No.	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
JIS 8	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
ID No.	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
JIS 8	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V

Protocol message structure

Send

The FB shown below is located in Net 8 within subroutine: Send command to Mitsubishi. Note how the messages in the Protocol Send FB are configured to conform to the Mitsubishi message structure.



ASCII character SOH(01) marks the beginning of a message

Station ID #, transmitted as hexadecimal

Command # (see table)

STX(02) preceeds data request

Type of data transmitted/ requested

The length and type of the data fields are determined by the command

ETX follows the data

End of Message determines checksum format and placement. Options are selected that conform to the protocol's requirements.

Select Name : MR-J2-10C-S100

Start Of Text (STX)

Length : 1

STX : 01

End Of Text (ETX)

Length : 1

ETX : 03

End Of Message: ETX_Checksum

Send Message number : MI 1008 : Snd: Message line number

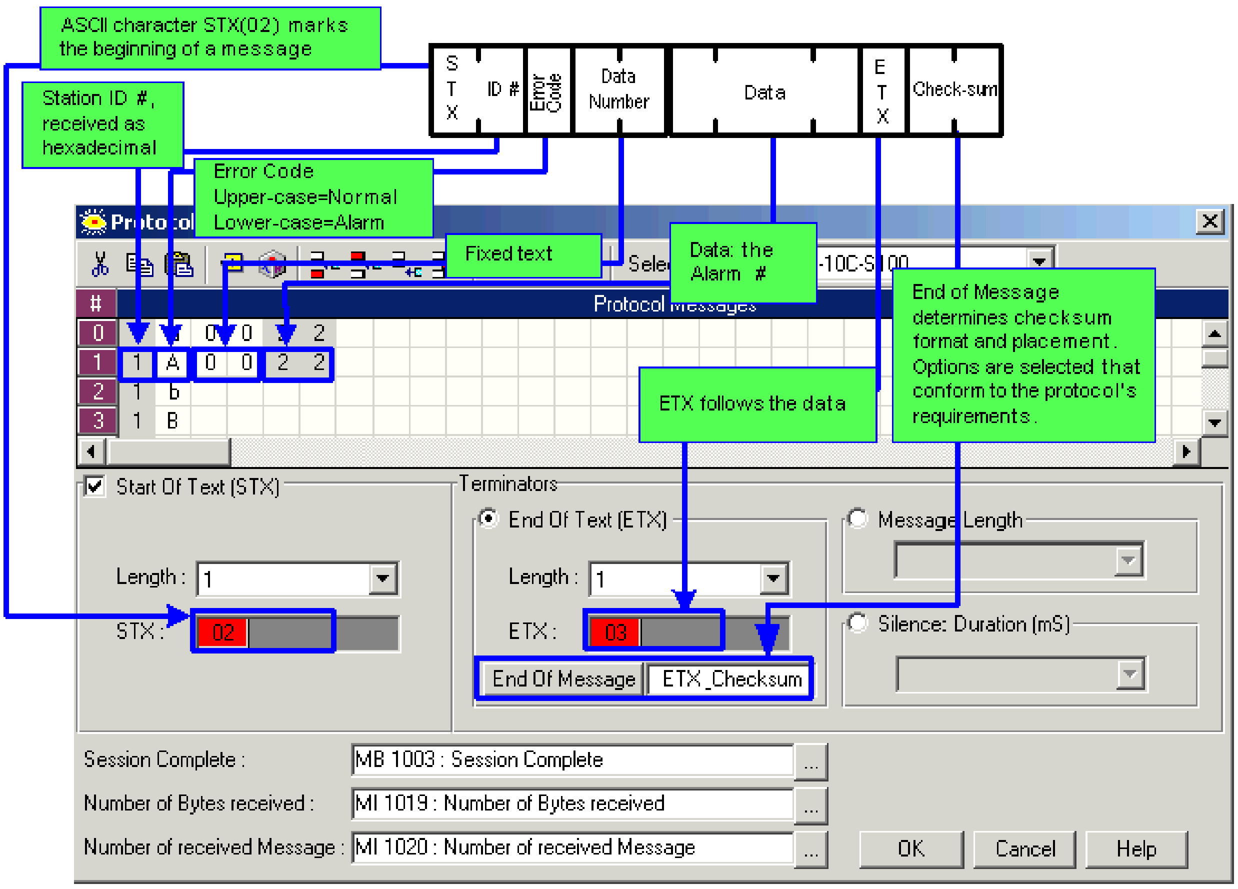
Scan

The FB shown below is located in Net 3 within subroutine: Check answer after sending. Note how the messages in the Protocol Send FB are created to conform to the Mitsubishi message structure.

Error codes

The Mitsubishi protocol uses an error code to determine if data is transmitted correctly.
When a converter receives a message, it replies by sending a message to the controller, according to the table below.

Error Code		
Normal	**Alarm**	**Description**
A	**a**	Data transmitted properly
B	**b**	Parity error
C	**c**	Checksum error
D	**d**	Character error
E	**e**	Command error
F	**f**	Data # error


ASCII character STX(02) marks the beginning of a message

Station ID #, received as hexadecimal

Error Code
Upper-case=Normal
Lower-case=Alarm

Fixed text

Data: the Alarm #

End of Message determines checksum format and placement. Options are selected that conform to the protocol's requirements.

ETX follows the data

#						
0	1	a	0	0	2	2
1	1	A	0	0	2	2
2	1	b				
3	1	B				

[✓] Start Of Text (STX)
Length : 1
STX : 02

Terminators
● End Of Text (ETX)
Length : 1
ETX : 03
End Of Message: ETX_Checksum

○ Message Length
○ Silence: Duration (mS)

Session Complete : MB 1003 : Session Complete
Number of Bytes received : MI 1019 : Number of Bytes received
Number of received Message : MI 1020 : Number of received Message

Protocol FB (TCP/IP)

Protocol FB (TCP/IP) Overview

Use the Communication Protocol TCP/IP operations located on the FBs menu to enable your controller to exchange data with TCP/IP-enabled devices, such as frequency converters, bar-code readers, and printers. The controller must comprise an Ethernet card. Communication Protocol TCP/IP operations are located on the FBs menu.

You can find general information about Ethernet, IP addressing, sockets, and ports in the topic About Ethernet.

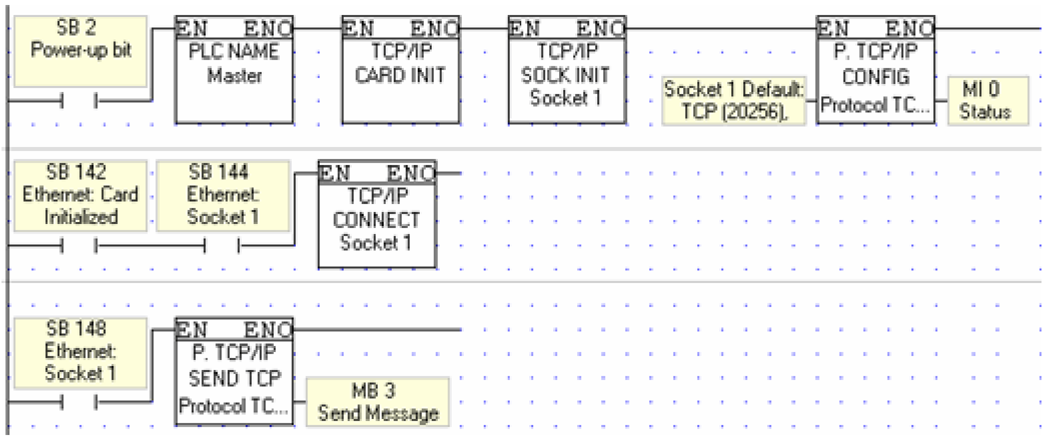
Specific information on implementing Ethernet is provided in the topic Using Ethernet.

Protocol TCP/IP operations are located on the FBs menu.

For more information regarding Protocol operations, refer to the topics below.

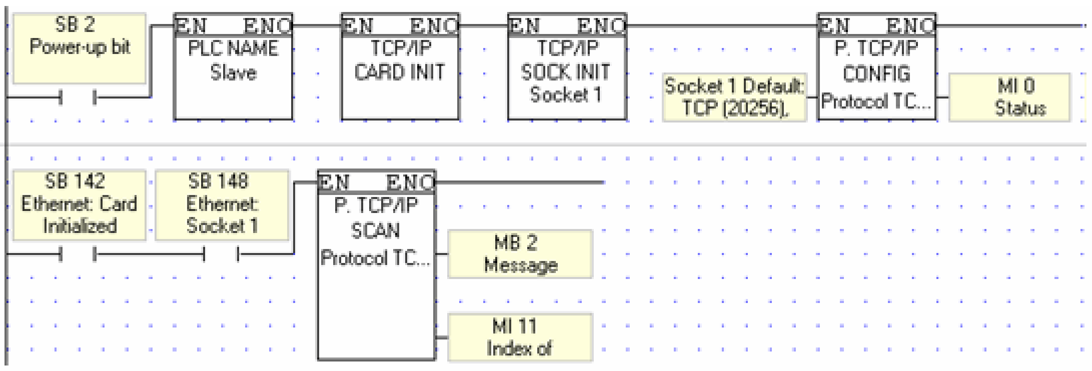
Example: TCP/IP Master

The nets below show the elements that are required by a master PLC. Note that the master application requires Socket Init and a TCP/IP Connect function, and that when you wish to terminate a session, you must include a Close Socket function.


SB 2 Power-up bit
EN ENO PLC NAME Master
EN ENO TCP/IP CARD INIT
EN ENO TCP/IP SOCK INIT Socket 1
Socket 1 Default: TCP (20256),
EN ENO P. TCP/IP CONFIG Protocol TC...
MI 0 Status
SB 142 Ethernet: Card Initialized
SB 144 Ethernet: Socket 1
EN ENO TCP/IP CONNECT Socket 1
SB 148 Ethernet: Socket 1
EN ENO P. TCP/IP SEND TCP Protocol TC...
MB 3 Send Message

Example: TCP/IP Slave

The nets below show the basic conditions enabling a TCP/ IP slave to receive messages.


SB 2 Power-up bit
EN ENO PLC NAME Slave
EN ENO TCP/IP CARD INIT
EN ENO TCP/IP SOCK INIT Socket 1
Socket 1 Default: TCP (20256),
EN ENO P. TCP/IP CONFIG Protocol TC...
MI 0 Status
SB 142 Ethernet: Card Initialized
SB 148 Ethernet: Socket 1
EN ENO P. TCP/IP SCAN Protocol TC...
MB 2 Message
MI 11 Index of

FB Operations

Configuration

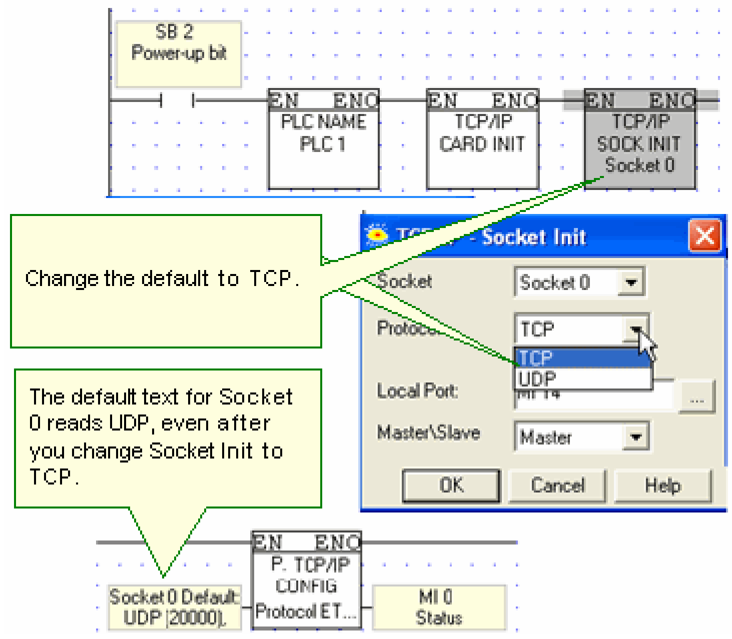
Scan

Send

To learn how the Scan operation works check the Protocol FB chapter. This chapter also includes information regarding Variables, Status and Error Messages.

Protocol TCP/IP: Configuration

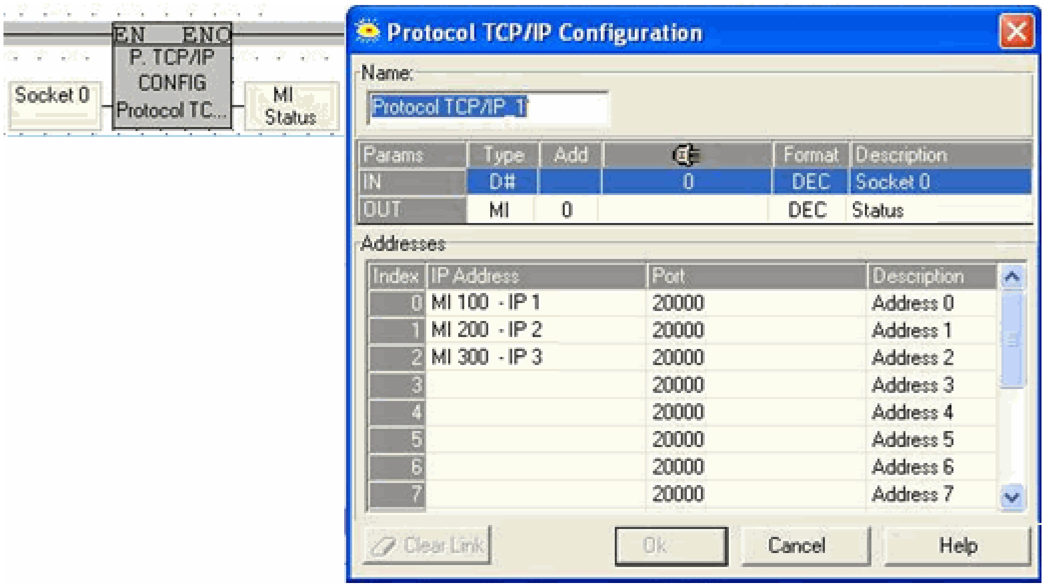
You create messages and attach variable data in Protocol TCP/IP Send and Scan operations. Each operation is linked to a Configuration. When a Send operation is called in your program, the Configuration determines which socket is used for outgoing messages; when a Scan operation is activated, the PLC monitors that socket for incoming messages.

Note •	The Configuration should be placed in the Main Routine, before any other FB operations. If the configuration is not active, Protocol operations will not be processed.
•	Before you place this FB in your Ladder, your application must include: - a Set PLC Name function - a TCP/IP Card Init to configure the Ethernet port - a Socket Init function, initializing a socket to TCP - TCP Connect and TCP Close functions
•	The Protocol FB does not support UDP. You can use Socket 0, if you change it to TCP in the Socket Init Function. Note that after you make this change, the default text remains UDP, even though the socket is initialized to TCP. 

Change the default to TCP.

The default text for Socket 0 reads UDP, even after you change Socket Init to TCP.

Socket: Socket 0
Protocol: TCP
Local Port:
Master\Slave: Master



Name: Protocol TCP/IP_1

Params	Type	Add		Format	Description
IN	D#		0	DEC	Socket 0
OUT	MI	0		DEC	Status

Addresses

Index	IP Address	Port	Description
0	MI 100 - IP 1	20000	Address 0
1	MI 200 - IP 2	20000	Address 1
2	MI 300 - IP 3	20000	Address 2
3		20000	Address 3
4		20000	Address 4
5		20000	Address 5
6		20000	Address 6
7		20000	Address 7

Parameter		**Function**
Name		Identifies the configuration. You use the name to link Protocol Send and Scan operations to a configuration.
(IN) Socket		Determines which socket is used for communicating the messages.
(OUT) Status	MI	If errors occur when the Configuration is used for a Send or Scan operation, the value of the linked MI indicates which error has occurred.
Addresses	MI or Constant	Provides the list of device's IP addresses for Scan operations.

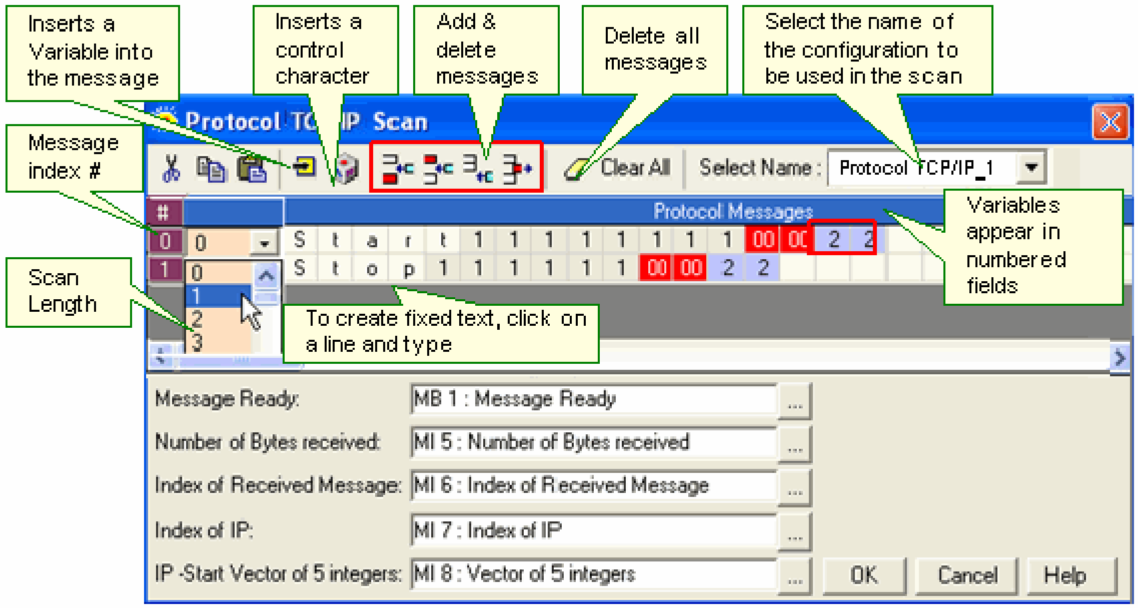
Protocol TCP/IP: Scan

A Scan operation contains messages. Activating the Scan causes the PLC to check if it has received any of the messages contained within the Scan. The PLC can only receive messages that are contained in a Scan FB.

Each Protocol Scan operation is linked to a particular Protocol Configuration. The Scan only registers messages that are received via the Socket defined in that Configuration.

Scan is located under FBs>Protocol TCP/IP.

Note •	Maximum message length is 500 characters; a message may include up to 50 variables. Note that, no section of a message--whether text or variable--may exceed 255 characters.
•	A message that is received overwrites a previously received message.
•	The PLC cannot receive more than 1 message per program scan; this means that if the PLC receiving messages has a scan cycle of 1 second, the sending device should not send more than 1 message per second.


Inserts a Variable into the message
Inserts a control character
Add & delete messages
Delete all messages
Select the name of the configuration to be used in the scan
Message index #
Scan Length
Variables appear in numbered fields
To create fixed text, click on a line and type
Protocol TCP/IP Scan
Clear All
Select Name : Protocol TCP/IP_1
Protocol Messages
OK
Cancel
Help

Message Ready: MB 1 : Message Ready
Number of Bytes received: MI 5 : Number of Bytes received
Index of Received Message: MI 6 : Index of Received Message
Index of IP: MI 7 : Index of IP
IP -Start Vector of 5 integers: MI 8 : Vector of 5 integers

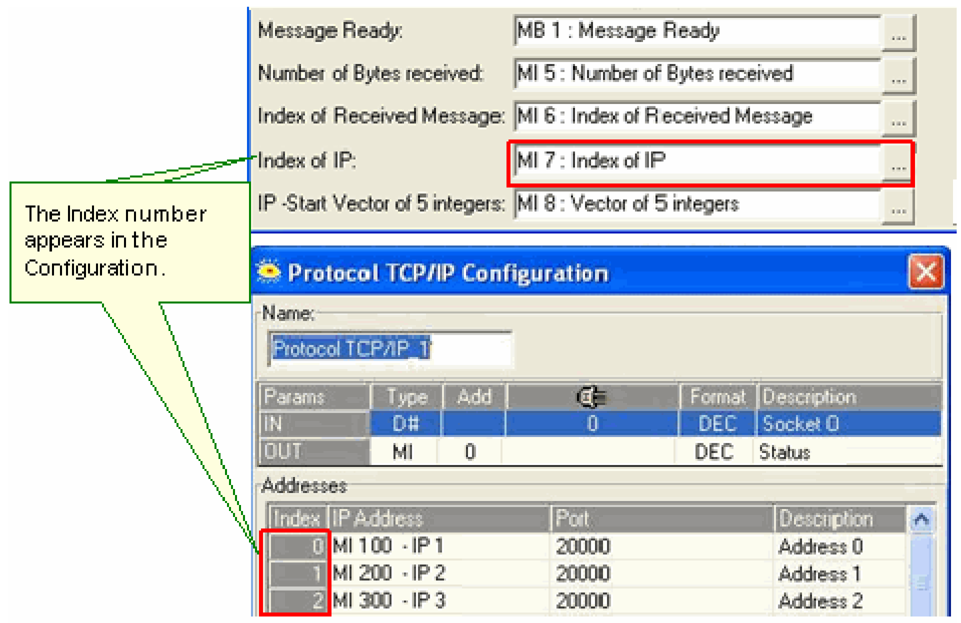
Message index	Identifies the incoming message.
Scan Length	This enables you to define the length of each incoming message, via the drop-down box that follows Message index #. Scan Length causes the Scan FB to use the length of a message to define whether an incoming message is legal. The length of an incoming message cannot be less than the length specified in the Scan operation. **Note •** If you use Scan Length, arrange the messages in ascending order, for example, message #0

		may contain 3 bytes, #1 may contain 4 bytes, #2 may contain 6 bytes, etc.

Paramete r		**Function**	
Message Received	M B	**Turns ON when:** Turns ON when the PLC receives a valid message.	**Turns OFF when** Turns OFF whenever the Scan operation is called by the program. In your program, link the Message Received bit to a positive contact, then use this condition to activate Copy operations, such as for the message variables, Number of Bytes Received and Number of Received Message.

Note •
-If the PLC receives a message which is **not** defined in the Scan FB, the message is invalid. The Message Received MB remains OFF.
-If a message in the Scan FB contains a Numeric Variable field', and the PLC receives this message with non-numerical characters in the field (except for leading spaces), the message is invalid. The Message Received MB remains OFF.
-An example of leading spaces is if, for example, the Numeric Variable field is 4 boxes long, and the 'number' 22 is received. In this case, the Scan FB will register the number22 preceded by 2 leading spaces (_ _ 2 2).

Number of Bytes Received	MI	This holds the number of message bytes. It is initialized by the system when a different message is received.
Index of Received Message	MI	This holds the index number of the received message. It changes when another message enters the system. The value in the linked MI is valid only when the Session Complete bit in ON. During this time, the linked MI contains the index number of the last valid incoming message. Once an invalid message has been received, the first incoming character of any message causes -1 to be written in the linked MI.
Index of IP	MI	When you create the Configuration, you create a list of IP addresses of the devices in your TCP/IP application. Each IP is assigned an index number. This list is used by the Scan operation. When a message is received, the IP and exit port of the sending device is written to a vector of 5 MIs. The first 4 contain the IP address, and the fifth will contain the port number. If the IP of the sending device matches an IP in the Configuration list, the index of the IP is written to this MI. **Note** • If the IP of the sending device is **not** in the Configuration list, the PLC receives the data, **but the value -1 is written to the MI**. .

		 **Note** • If the PLC receives a message from a device with an IP that is not in the Configuuse Scan Length, arrange the messages in ascending order, for example, message #0 may contain 3 bytes, #1 may contain 4 bytes, #2
IP Start of Vector, 5 Integers	MI	This is the start address of a vector of 5 MIs. When a message is received, the IP and exit port of the sending device is written to this vector The first 4 contain the IP address, and the fifth will contain the port number.

Message Ready: MB 1 : Message Ready
Number of Bytes received: MI 5 : Number of Bytes received
Index of Received Message: MI 6 : Index of Received Message
Index of IP: MI 7 : Index of IP
IP -Start Vector of 5 integers: MI 8 : Vector of 5 integers

Protocol TCP/IP Configuration
Name: Protocol TCP/IP_1

Params	Type	Add		Format	Description
IN	D#		0	DEC	Socket 0
OUT	MI	0		DEC	Status

Addresses

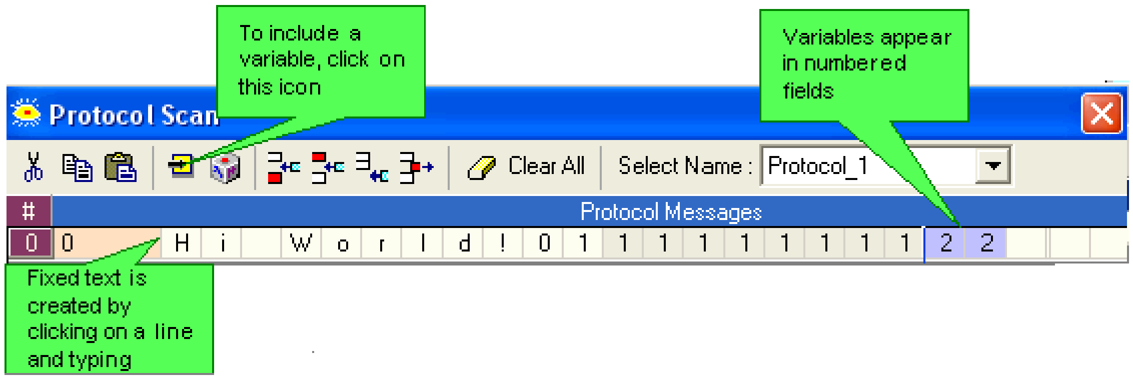
Index	IP Address	Port	Description
0	MI 100 - IP 1	20000	Address 0
1	MI 200 - IP 2	20000	Address 1
2	MI 300 - IP 3	20000	Address 2

Protocol Messages

Messages can include:

- **Fixed Text**

This text does not change.


To include a variable, click on this icon
Variables appear in numbered fields
Protocol Scan
Clear All
Select Name : Protocol_1
Protocol Messages
Fixed text is created by clicking on a line and typing

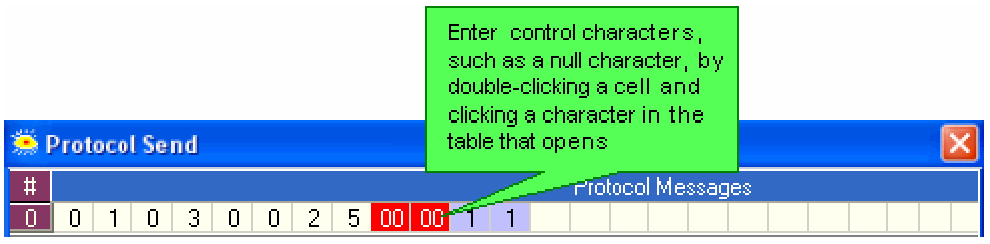
#																						
0	0	H	i		W	o	r	l	d	!	0	1	1	1	1	1	1	1	1	1	2	2

- **Variables**

These variables may be received by the PLC: Numeric, Stream, and Checksum.

- **Control Characters**

These can be added according to protocol requirements.

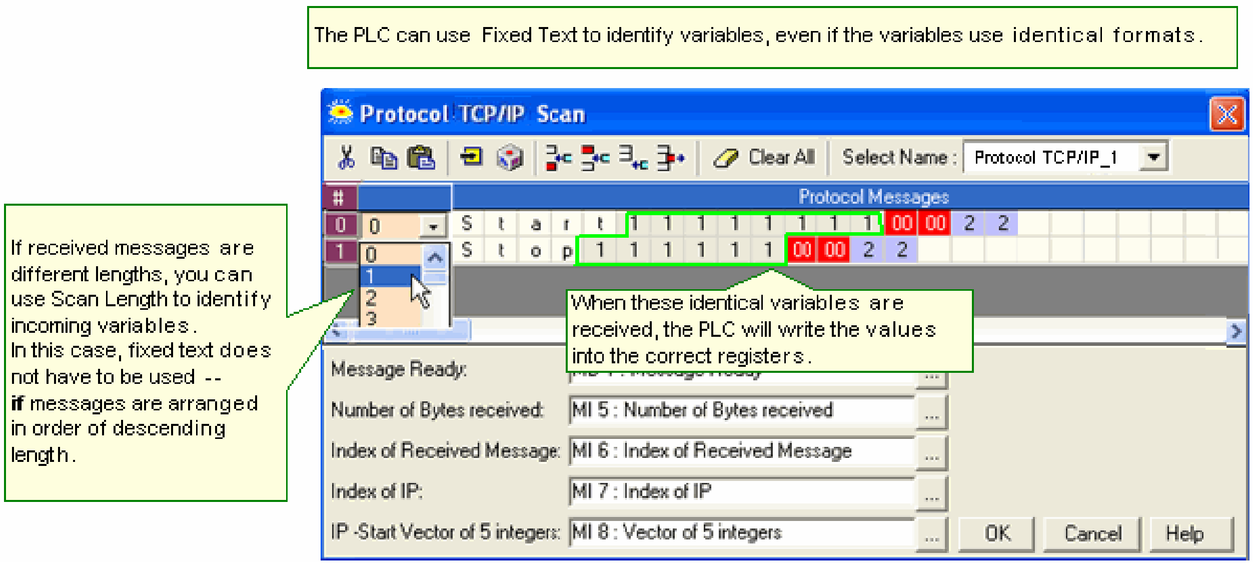

Protocol Send
Enter control characters, such as a null character, by double-clicking a cell and clicking a character in the table that opens
Protocol Messages

#	Protocol Messages											
0	0	1	0	3	0	0	2	5	00	00	1	1

Scan operations containing more than one message

Note that if a Scan operation contains more than one message with identical variables that use the same format, the PLC will overwrite the variable values as they are received.

To enable the PLC to differentiate between the messages and their attached variables, include fixed text and/or control characters within the message.

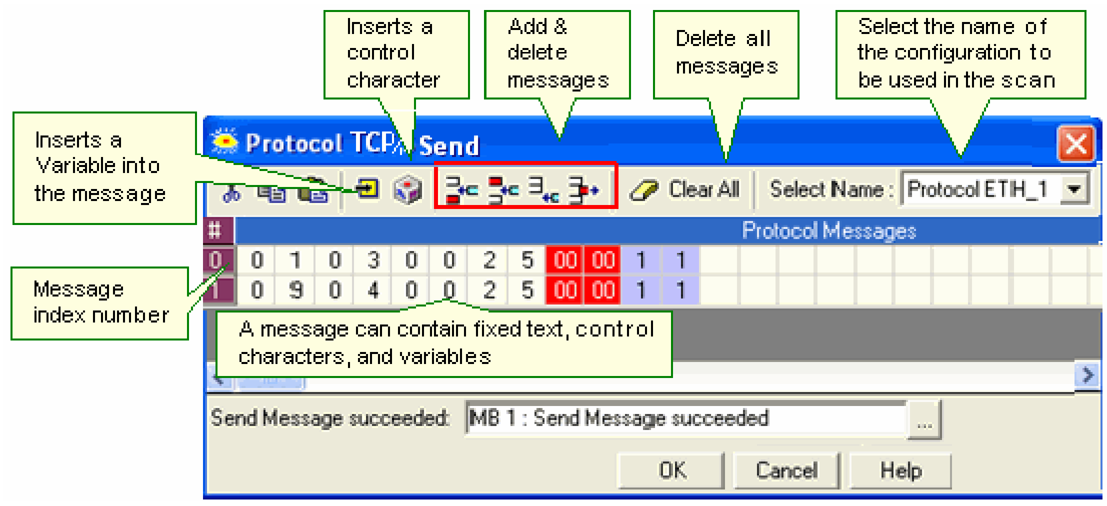

The PLC can use Fixed Text to identify variables, even if the variables use identical formats.
If received messages are different lengths, you can use Scan Length to identify incoming variables. In this case, fixed text does not have to be used -- if messages are arranged in order of descending length.
When these identical variables are received, the PLC will write the values into the correct registers.
Protocol TCP/IP Scan
Clear All
Select Name : Protocol TCP/IP_1
Protocol Messages
OK
Cancel
Help

#		Protocol Messages																
0	0	S	t	a	r	t	1	1	1	1	1	1	1	1	00	00	2	2
1	0	S	t	o	p	1	1	1	1	1	1	00	00	2	2			

Message Ready: MI 4 : Message Ready
Number of Bytes received: MI 5 : Number of Bytes received
Index of Received Message: MI 6 : Index of Received Message
Index of IP: MI 7 : Index of IP
IP -Start Vector of 5 integers: MI 8 : Vector of 5 integers

Protocol TCP/IP: Send

Each Send operation is linked to a particular Configuration; outgoing messages are sent via the COM port selected in that Configuration. Send is located on the FBs menu.


Inserts a control character
Add & delete messages
Delete all messages
Select the name of the configuration to be used in the scan
Inserts a Variable into the message
Message index number
A message can contain fixed text, control characters, and variables
Protocol TCP/IP Send
Clear All
Select Name :
Protocol Messages
OK
Cancel
Help

Select Name : Protocol ETH_1

#												
0	0	1	0	3	0	0	2	5	00	00	1	1
1	0	9	0	4	0	0	2	5	00	00	1	1

Send Message succeeded: MB 1 : Send Message succeeded

Send Message Number	Determines which message in the FB will be sent. In the program, store a message index numbers in the linked register. When the FB is called, message whose index number currently in the linked MI will be sent.

Protocol Messages

Messages can include:

- **Fixed Text**

Fixed text must be included if there is more than one message in a Scan. This enables the PLC to identify the message.

- **Control Characters**

These can be added according to protocol requirements.

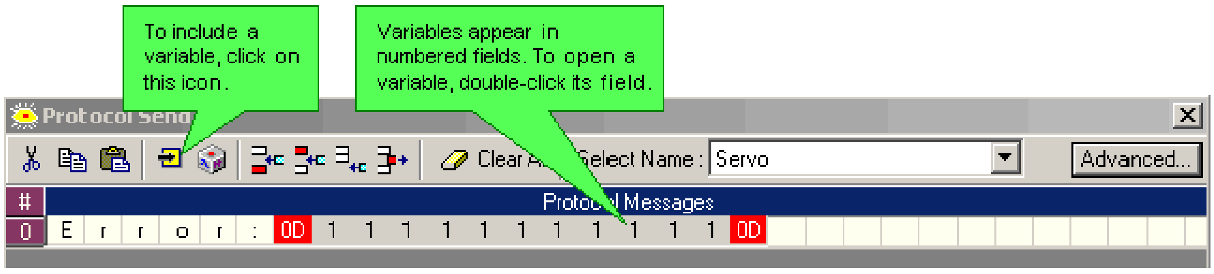
- **Variables**

These variables may be sent by the PLC: Binary, Numeric, Stream, List of Texts, Checksum, Date, and Time.

Variables

Variables enable you to send and receive a variety of dynamic data from your process.

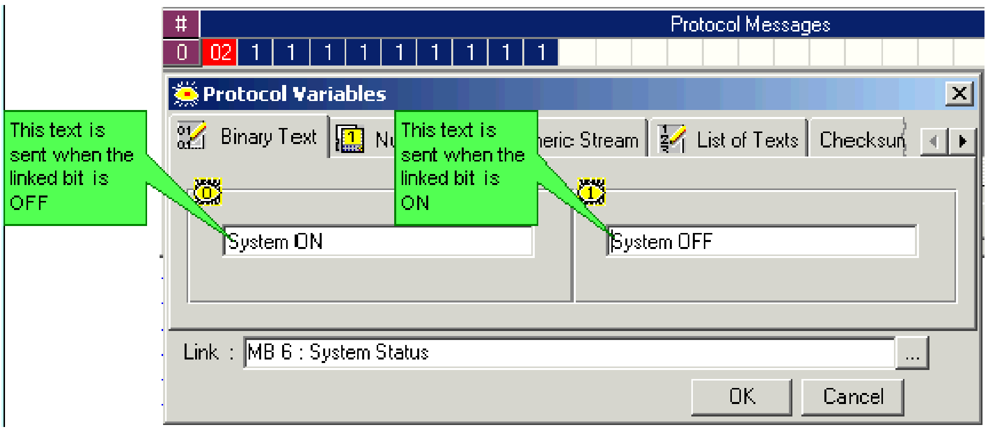
Variable Types: Send


To include a variable, click on this icon.
Variables appear in numbered fields. To open a variable, double-click its field.
Protocol Send
Clear All
Select Name : Servo
Advanced...
Protocol Messages
0 E r r o r : 0D 1 1 1 1 1 1 1 1 1 1 1 0D

The PLC sends messages via the Configuration linked to the Send operation. These are the types of variables that can be attached to an outgoing message.

- **Binary text**

Use this type of variable to send text based on the status of a linked bit.



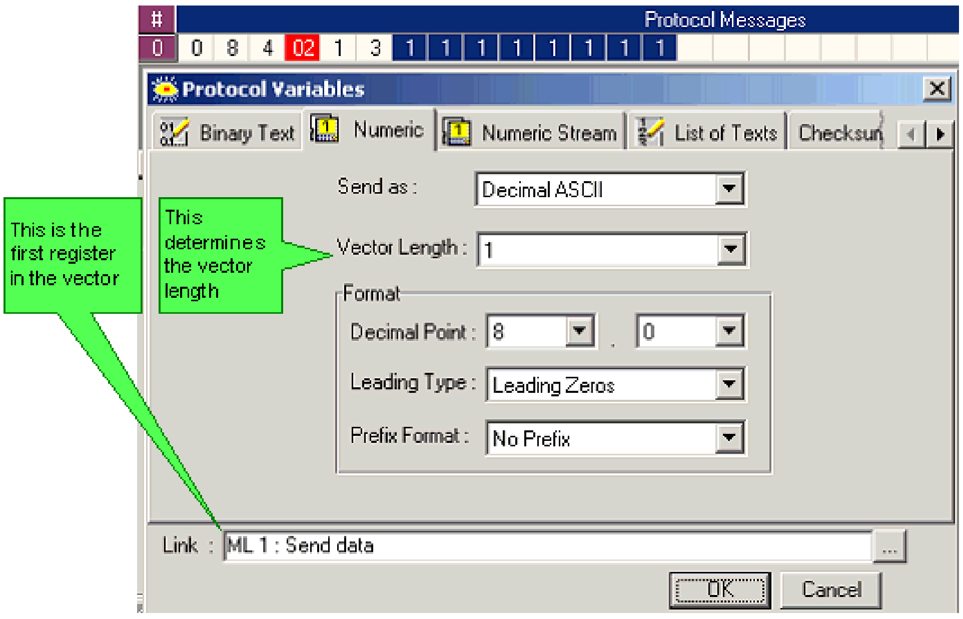
- **Numeric**

Use a Numeric variable to send a vector of registers. You can send register values as Binary, Hex as ASCII, or Decimal ASCII. The variable field automatically adjusts to fit the size of the data to be sent.

Note • Sending Hex as ASCII: the PLC transfers the value to ASCII, and sends it byte by byte, beginning from the most significant value.

Thus, the value 8F03BCA1 sent "Hex as ASCII" will be converted to Hex and sent as follows:

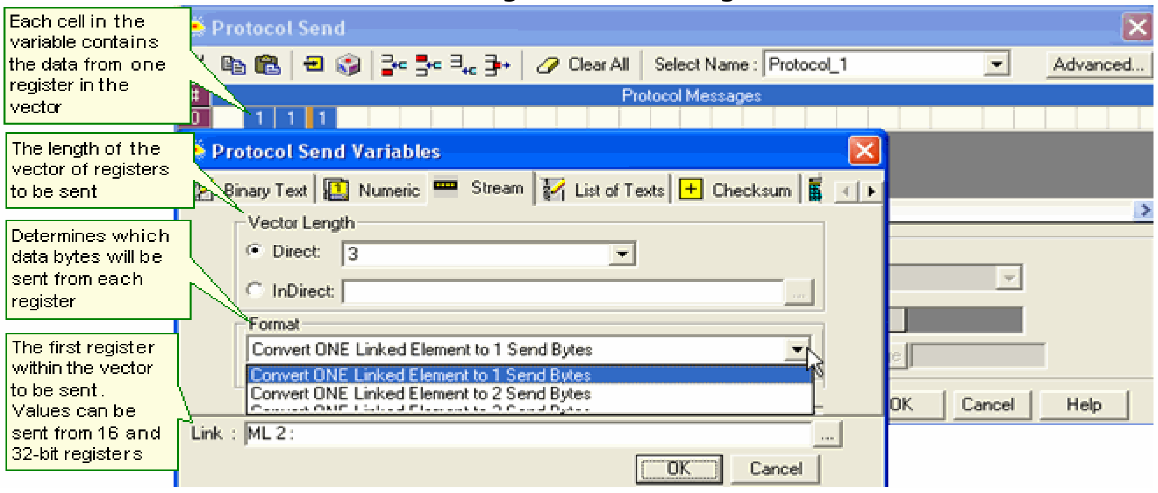
38, 46, 30, 33, 42, 43, 41, 31.



Send as : Decimal ASCII
Vector Length : 1
Format
Decimal Point : 8 . 0
Leading Type : Leading Zeros
Prefix Format : No Prefix
Link : ML 1 : Send data

- **Stream**

Use Stream to send a vector of registers to a target device.

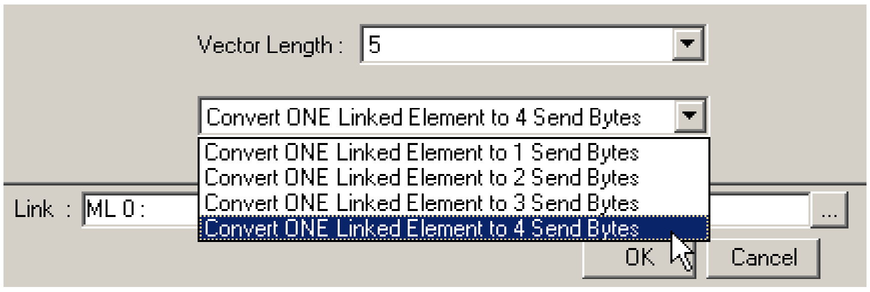

Protocol Send
Clear All
Advanced...
Protocol Messages
Protocol Send Variables
Binary Text
Numeric
Stream
List of Texts
Checksum
Each cell in the variable contains the data from one register in the vector
The length of the vector of registers to be sent
Determines which data bytes will be sent from each register
The first register within the vector to be sent. Values can be sent from 16 and 32-bit registers
OK
Cancel
Help

Select Name : Protocol_1
Vector Length
Direct: 3
InDirect:
Format
Convert ONE Linked Element to 1 Send Bytes
Convert ONE Linked Element to 1 Send Bytes
Convert ONE Linked Element to 2 Send Bytes
Link : ML 2 :

Note •	Values are sent in binary format, beginning from the LSB. Thus, the value 8F03BCA1 is sent as follows: A1, BC, 3, 8F.
•	In order to send floating values, select MF and the Convert 4 bytes into 1 linked element option. Note that the PLC uses the big-endian system; meaning that the most significant value in the sequence is stored at the lowest (first)storage address).

Stream Variable Format

The format determines which bytes are sent from within a register.


Vector Length : 5

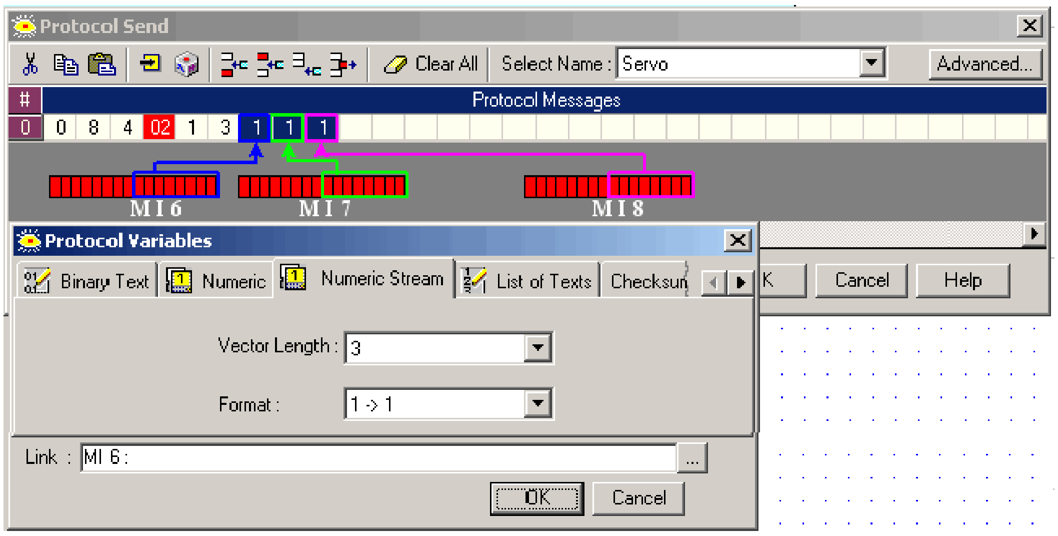
Convert ONE Linked Element to 4 Send Bytes

Convert ONE Linked Element to 1 Send Bytes
Convert ONE Linked Element to 2 Send Bytes
Convert ONE Linked Element to 3 Send Bytes
Convert ONE Linked Element to 4 Send Bytes

Link : ML 0 :

- **1->1**

Select this to send the low byte (first 8 bits) of each register in the vector.



#									
0	0	8	4	02	1	3	1	1	1

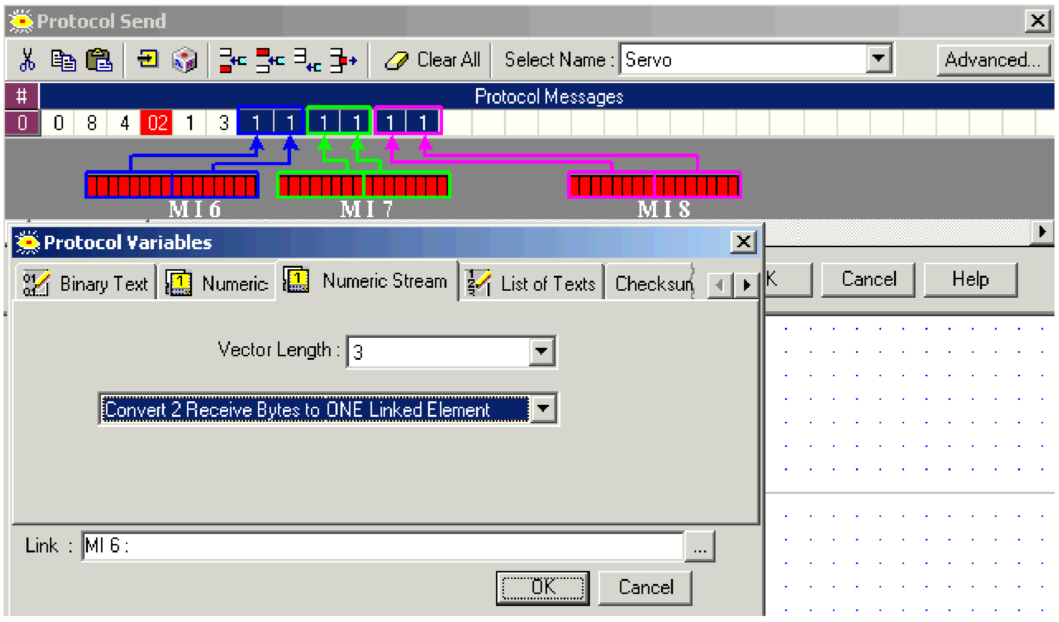
Vector Length : 3

Format : 1 -> 1

Link : MI 6 :

- **2->1**

Select this to send both low and high bytes from each register in the vector. Note that this is an entire 16-bit register, and the first 2 bytes of a 32-bit register.



#												
0	0	8	4	02	1	3	1	1	1	1	1	1

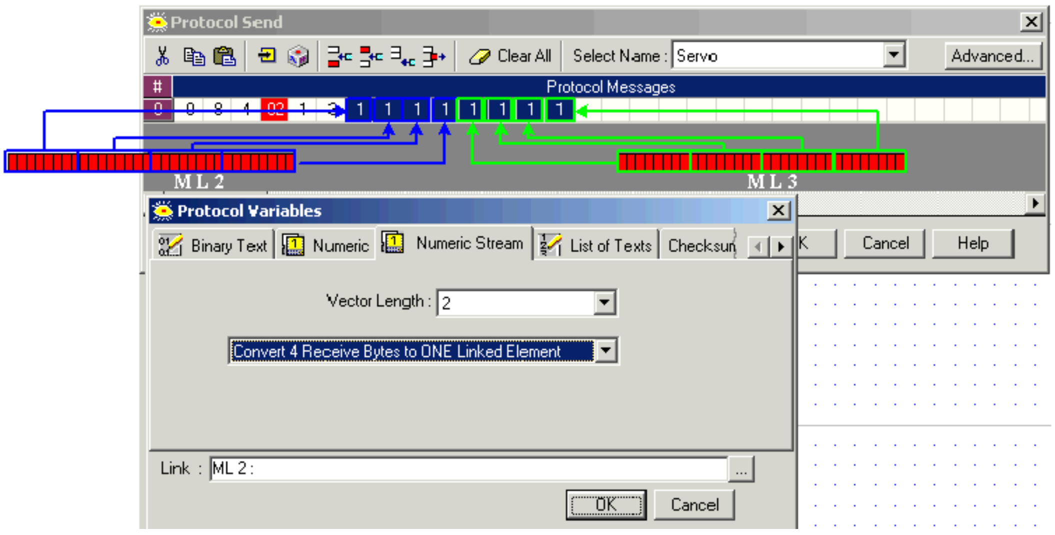
Vector Length : 3

Convert 2 Receive Bytes to ONE Linked Element

Link : MI 6 :

- **4->1**

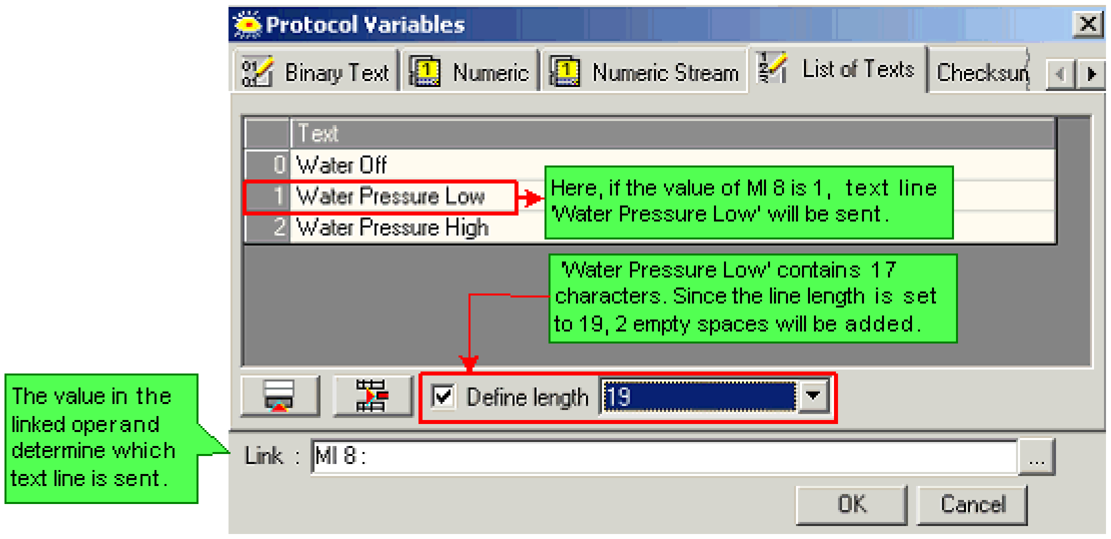
Select this to send all 4 bytes of each 32-bit register in a vector.



- **List of texts**

This type of variable contains a list of numbered lines of text. The value within the linked operand 'points' to the number of a line within the list. When the operand value is equal to a particular line number, the text of that line is included in the variable.

Selecting the Define Length option enables you to make all of the text lines a uniform length, regardless of the number of characters a specific line may have. If a text line is shorter than the defined length, empty spaces will be added; if the text line is longer than the defined length, excess characters will be cut off.

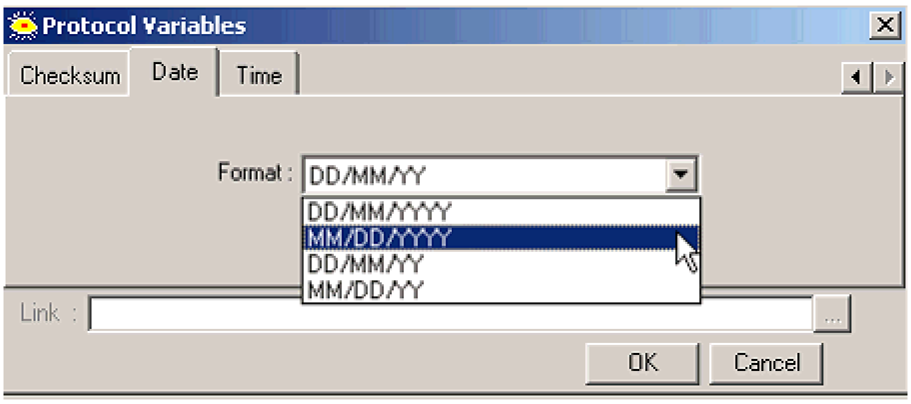

Protocol Variables
Binary Text
Numeric
Numeric Stream
List of Texts
Checksum
Text
0 Water Off
1 Water Pressure Low
2 Water Pressure High
Here, if the value of MI 8 is 1, text line 'Water Pressure Low' will be sent.
'Water Pressure Low' contains 17 characters. Since the line length is set to 19, 2 empty spaces will be added.
The value in the linked operand determine which text line is sent.
Define length 19
Link : MI 8 :
OK
Cancel

- **Checksum**

Via this variable, you can calculate a checksum and attach it to a message. Use the options to configure the checksum according to the requirements of your particular protocol.

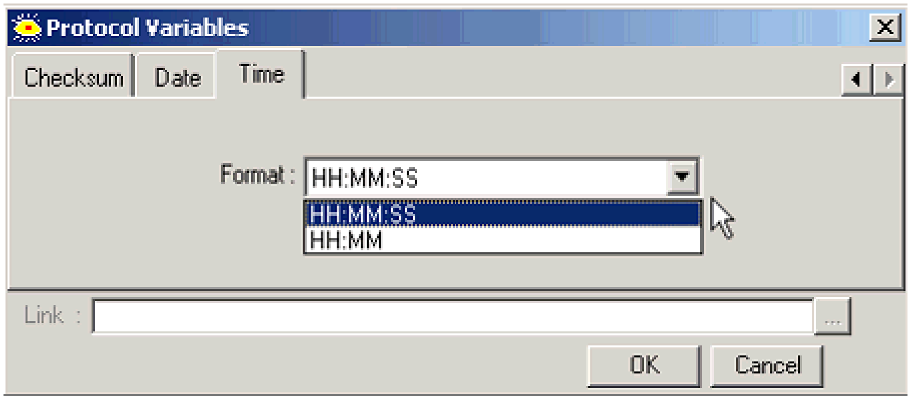
- **Date**

Use this variable to send the current date according to a selected format.



- **Time**

Use this variable to send the current time according to a selected format.



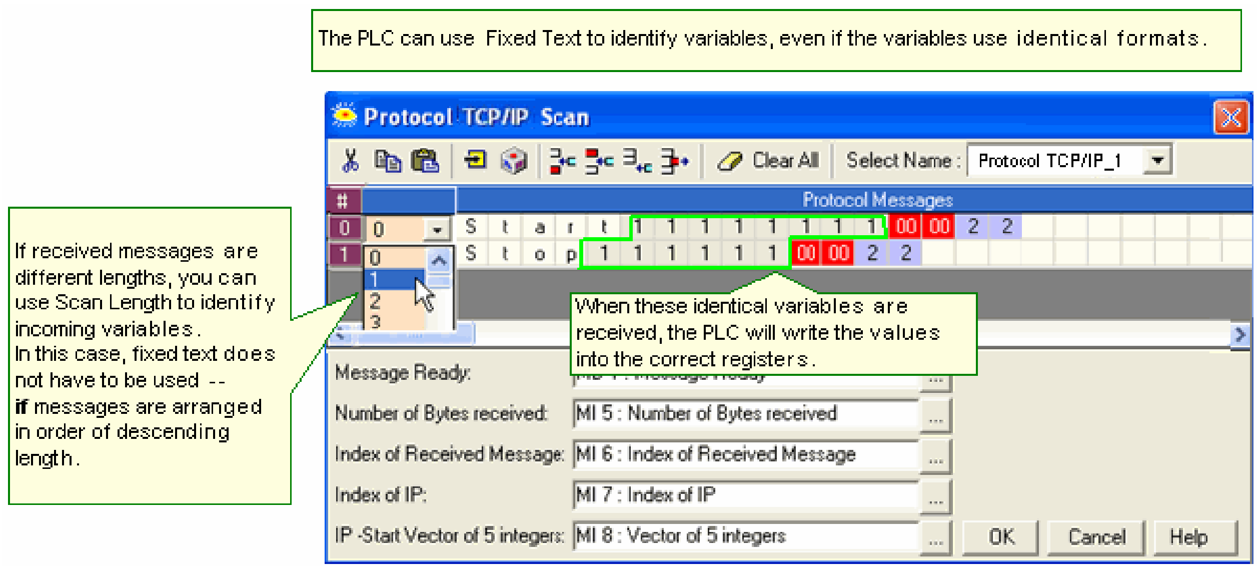
Variable Types: Scan

The PLC scans for messages that enter the system via the Configuration linked to the Scan operation. These are the types of variables that can be attached to an incoming message.

To correctly identify incoming variables, use either Scan Length, or include fixed text.

Notes •	Scan Length can be defined for each message via the drop-down box that follows Message index #. Scan length causes the Scan FB to use the length of a message to define whether an incoming message is legal. This parameter overrides Start Of Text and Terminator settings. If you use Scan Length, arrange the messages in ascending order, for example, message #0 may contain 3 bytes, #1 may contain 4 bytes, #2 may contain 6 bytes, etc.
•	If a Scan operation contains more than one message with identical variables that use the same format, the PLC will overwrite the variable values as they are received.

To enable the PLC to differentiate between the messages and their attached variables, include fixed text and/or control characters within the message.


The PLC can use Fixed Text to identify variables, even if the variables use identical formats.

If received messages are different lengths, you can use Scan Length to identify incoming variables.
In this case, fixed text does not have to be used --
if messages are arranged in order of descending length.

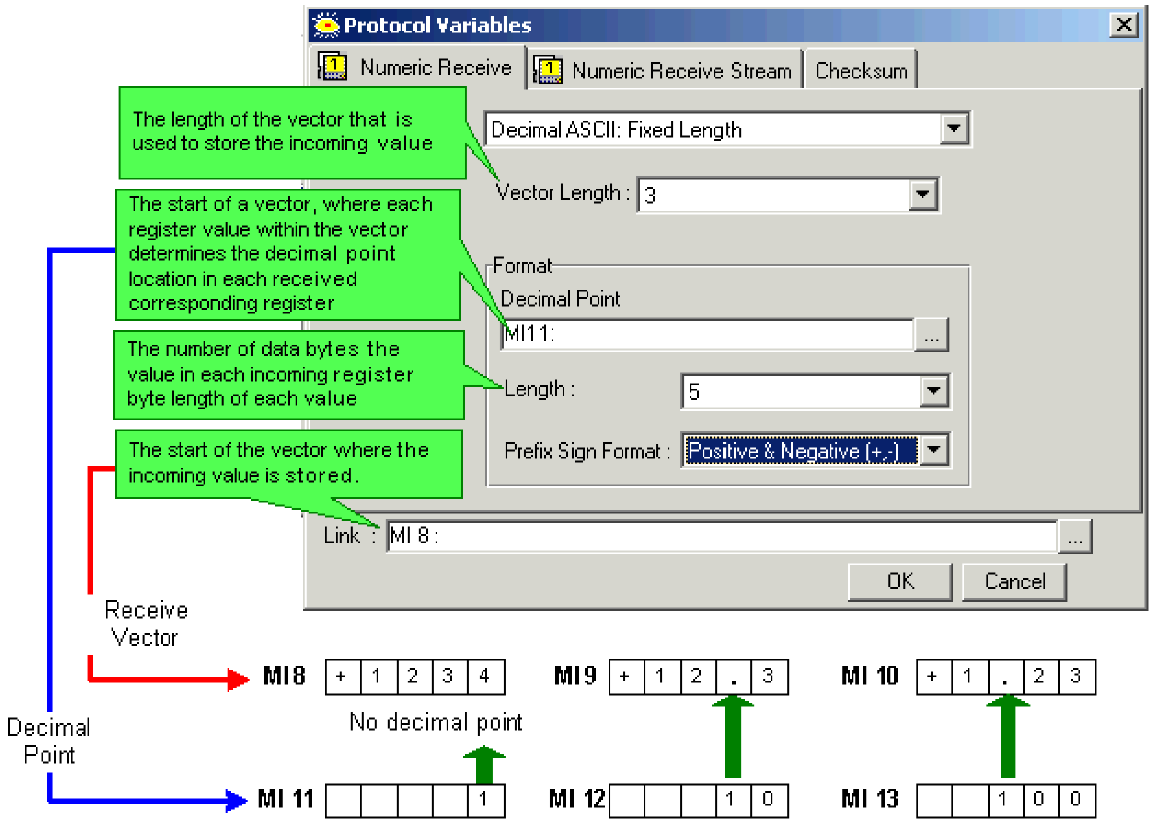
When these identical variables are received, the PLC will write the values into the correct registers.

Protocol TCP/IP Scan
Clear All
Select Name : Protocol TCP/IP_1
Protocol Messages
S t a r t 1 1 1 1 1 1 1 1 00 00 2 2
S t o p 1 1 1 1 1 1 00 00 2 2

Message Ready:	MB 1 : Message Ready
Number of Bytes received:	MI 5 : Number of Bytes received
Index of Received Message:	MI 6 : Index of Received Message
Index of IP:	MI 7 : Index of IP
IP -Start Vector of 5 integers:	MI 8 : Vector of 5 integers

- **Numeric Receive**

Use a Numeric variable to receive a vector of registers. You can receive register values as Binary, Hex as ASCII, or Decimal ASCII.


The length of the vector that is used to store the incoming value

The start of a vector, where each register value within the vector determines the decimal point location in each received corresponding register

The number of data bytes the value in each incoming register byte length of each value

The start of the vector where the incoming value is stored.

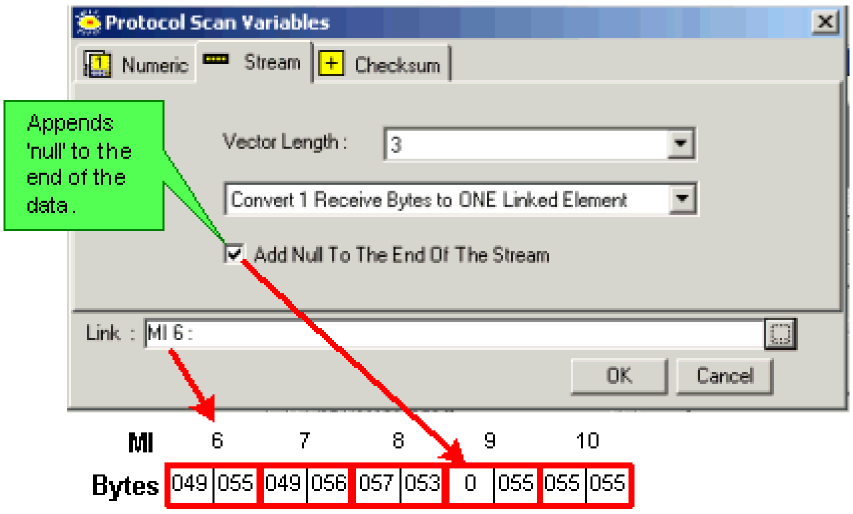
Decimal ASCII: Fixed Length
Vector Length : 3
Format
Decimal Point: MI 11 :
Length : 5
Prefix Sign Format : Positive & Negative (+,-)
Link : MI 8 :

- **Receive Stream**

Use Stream to receive a vector of register values from an external device and copy them into a defined vector of registers within the PLC.

Note • A vector is read either until the end of the defined vector length, or until a null character is encountered. By adding a null character to the end of the stream, you can mark the end of a data string. This

	can prevent other data, that might be present in a vector, from being added to the data string when the vector is read.
	In order to send floating values, select MF and the Convert 4 bytes into 1 linked element option. Note that the PLC uses the big-endian system; meaning that the most significant value in the sequence is stored at the lowest (first)storage address).


Protocol Scan Variables
Numeric
Stream
Checksum
Appends 'null' to the end of the data.
OK
Cancel
MI 6 7 8 9 10
Bytes 049 055 049 056 057 053 0 055 055 055

Vector Length : 3

Convert 1 Receive Bytes to ONE Linked Element

[✓] Add Null To The End Of The Stream

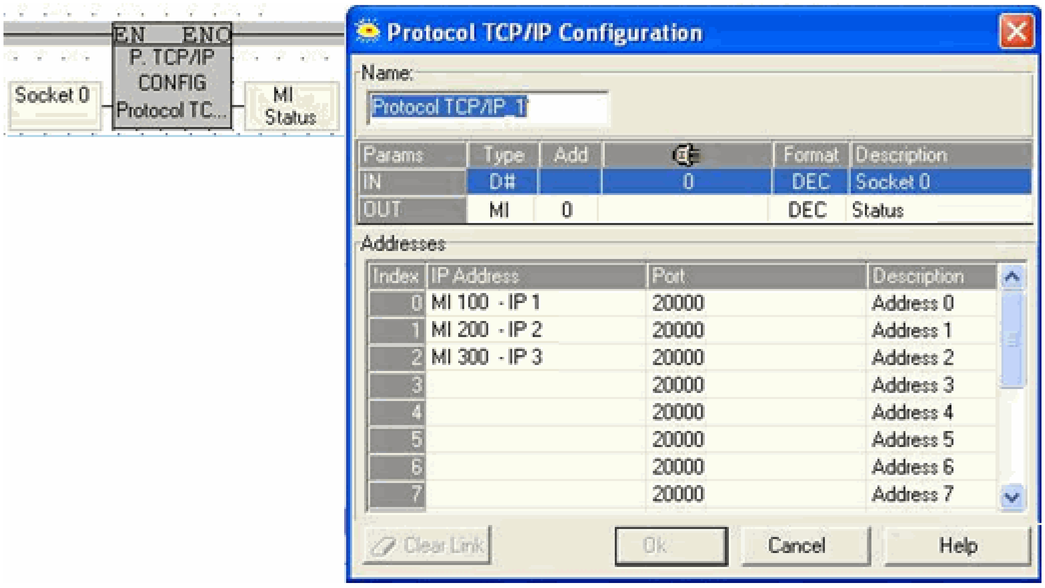
Link : MI 6 :

- **Checksum**

Via this variable, you can process the checksum of an incoming message. Use the options to configure the checksum according to the requirements of your particular protocol.

Protocol TCP/IP Status Messages

The MI Status Operand in the Configuration shows the state of Protocol communications.



Name: Protocol TCP/IP_1

Params	Type	Add		Format	Description
IN	D#		0	DEC	Socket 0
OUT	MI	0		DEC	Status

Addresses

Index	IP Address	Port	Description
0	MI 100 - IP 1	20000	Address 0
1	MI 200 - IP 2	20000	Address 1
2	MI 300 - IP 3	20000	Address 2
3		20000	Address 3
4		20000	Address 4
5		20000	Address 5
6		20000	Address 6
7		20000	Address 7

Protocol Operation Status Operands

When you place a Protocol Configuration your application, the linked operands indicate the status of Protocol operations.

Status Messages	MI	Automatically initialized to 0 when an operation is activated. Updated at the end of each attempt to communicate via the Configuration. Indicates status of communications, according to the table below. Note that the current value always shows the most **recent** status.

Value	**Error Message**	**Indication**
0	No Error	Operations successful
2	General Configuration Error	Indicates an unknown system error.
257	Send: General Error	Indicates an unknown system error.
259	Send: Invalid Message Index	The message that is referenced in the Send operation does not exist.
260	Send: Invalid Operand Type	The message references an operand type that does not exist in the system.
261	Send: Invalid Operand Address	The message references an operand address that does not exist in the system.
262	Send: Data Overflow	The message data exceeds 512 bytes.
263	Send: Format Error	The message contains variable fields that are not long enough for the

		attached variable data.
264	Send: TCP/IP General Error	Check the TCP/ IP SIs
513	Receive: General Error	Indicates an unknown system error.
514	Receive: Invalid message	The received message does not exist in the system.
515	Receive: Checksum Error	The message may have been corrupted during transmission.

Note •	To automatically reconnect a lost Ethernet connection, turn SB 168 Automatically reconnect (keep alive) ON at power-up.
•	To enable PLC to disconnect when the Ethernet connection is inactive for a period of time, assign timeout values in SIs 103 -106
•	To enable a PLC attempt to reconnect when there is no communication from the connected device for a period of time, assign timeout values in SIs 107 -110

PID

PID Overview

The PID FB enables you to use system feedback to continuously control a dynamic process. The purpose of PID control is to keep a process running as close as possible to a desired Set Point. VisiLogic's PID FB includes auto-tune.

In order to ensure proper PID function, you should use Autotune.

During the Autotune process, the PID function collects certain essential data. Unitronics' proprietary PID algorithm uses this data to run smooth, accurate PID.

Note that you can use the Read Control Components FB in conjunction with the PID Server utility to create a data file to send to support@unitronics in the event that you require technical support for PID applications. Information on how to create a data file is in the PID Server help file.

FB Operations

PID Configuration

Run Auto Tune

Run PID

Pause Integral & Derivative Calculation

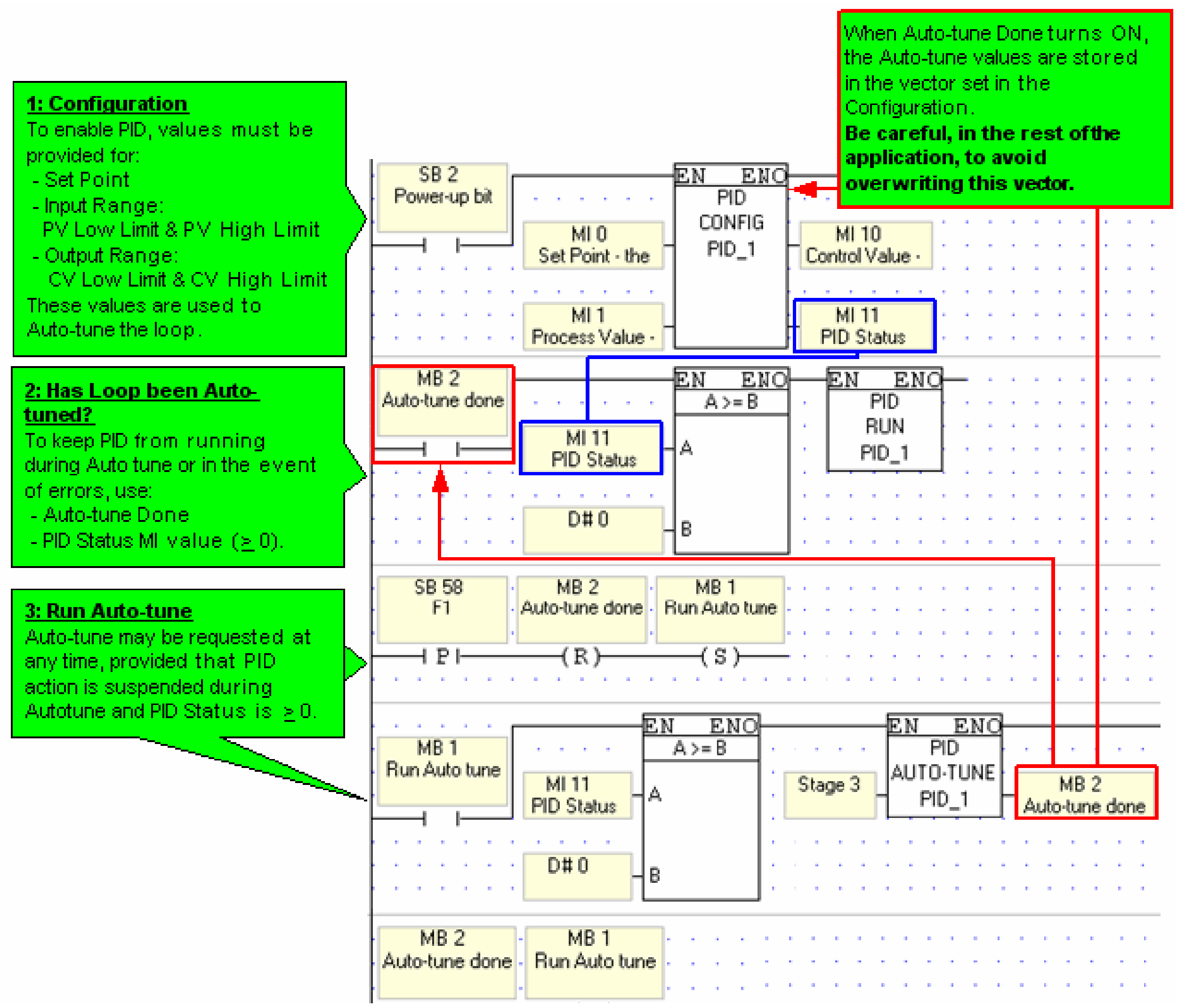
Error Integral

Read Control Components

PID Autotune

Auto-tuning loops enables the system to set the parameters for PID.

The picture below shows the elements of a basic PID application with Auto-tune.



1: Configuration
To enable PID, values must be provided for:
- Set Point
- Input Range:
  PV Low Limit & PV High Limit
- Output Range:
  CV Low Limit & CV High Limit

These values are used to Auto-tune the loop.

2: Has Loop been Auto-tuned?
To keep PID from running during Auto tune or in the event of errors, use:
- Auto-tune Done
- PID Status MI value (≥ 0).

3: Run Auto-tune
Auto-tune may be requested at any time, provided that PID action is suspended during Autotune and PID Status is ≥ 0.

When Auto-tune Done turns ON, the Auto-tune values are stored in the vector set in the Configuration.
Be careful, in the rest of the application, to avoid overwriting this vector.

After Auto-tune runs, the P, I, and D values are automatically written to the Configuration parameters and the Auto-tune vector is also filled with the Auto-tune parameters.

Note
•

Note that, once you have run Auto-tune, you can back up the P, I, and D values, the sample time (ST), and the 32 MI-long Auto-tune vector into a Data Table. You can then transfer these values to another Vision controlling an identical system, in order to run PID without tuning the loop.

Examples

PID Tips can help you with certain PID applications, such as those using SSR switches.

PID Configuration

To place a PID Configuration:

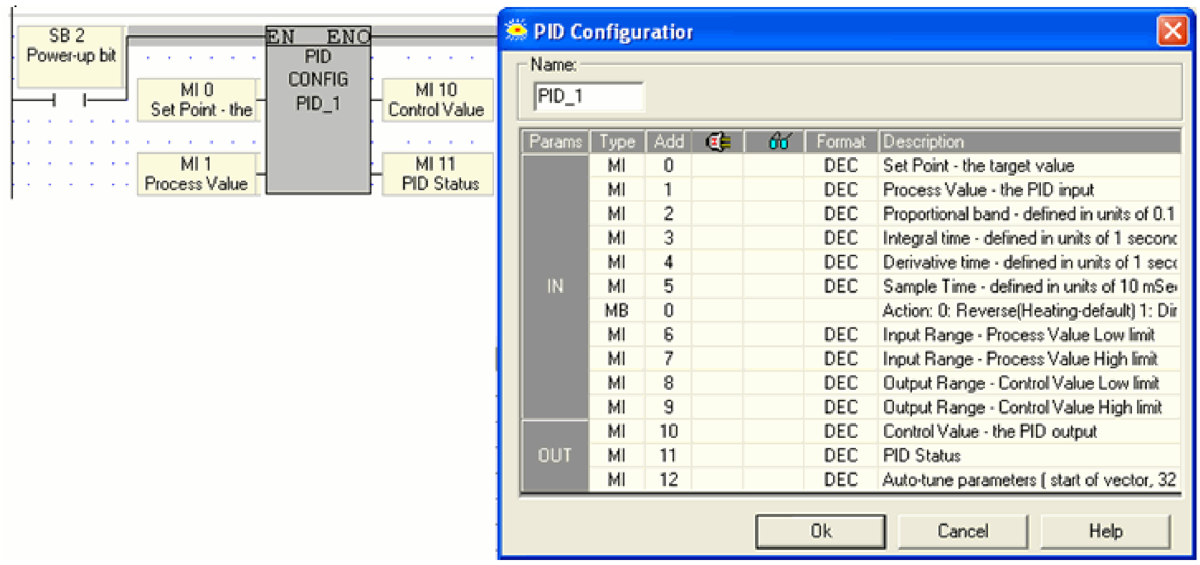
1. Select PID Configuration from the FBs menu, then place the function in the net; the PID parameter box opens.
2. The Select Operand and Address box opens; prompting you to link operands to the PID parameters.

Note • To enable PID, values must be provided for:

- Set Point
- Input Range:PV Low Limit & PV High Limit
- Output Range:CV Low Limit & CV High Limit

These values are used to Auto-tune the loop.

After Auto-tune runs, the P, I, D and Sample Time values are automatically written to the Configuration parameters.



Name: PID_1

Params	Type	Add			Format	Description
IN	MI	0			DEC	Set Point - the target value
	MI	1			DEC	Process Value - the PID input
	MI	2			DEC	Proportional band - defined in units of 0.1
	MI	3			DEC	Integral time - defined in units of 1 seconc
	MI	4			DEC	Derivative time - defined in units of 1 secc
	MI	5			DEC	Sample Time - defined in units of 10 mSe
	MB	0				Action: 0: Reverse(Heating-default) 1: Dir
	MI	6			DEC	Input Range - Process Value Low limit
	MI	7			DEC	Input Range - Process Value High limit
	MI	8			DEC	Output Range - Control Value Low limit
	MI	9			DEC	Output Range - Control Value High limit
OUT	MI	10			DEC	Control Value - the PID output
	MI	11			DEC	PID Status
	MI	12			DEC	Auto-tune parameters [start of vector, 32

PID Function Parameters

Parameters: Inputs	**Type**	**Function**
SP: Set Point	MI	SP is the target value for the process. In a heating system, this is the temperature value set for the system. Note that the Set Point and Process value must be given in the same type of units (degrees Celsius, bars, meters per second, etc.)
PV: Process Value	MI	PV is the feedback from the process. PV is output from the process and input to the PID function. In a heating system, the temperature measured by a temperature sensor provides the PV.
Kp: Proportional Band	MI	Use this parameter to define the proportional band, in units of 0.1%. The proportional band is a percentage of the total Process Value (PV). It is a range defined around the Set Point. When the PV is within this range, the PID function is active.
Ti: Integral Time	MI	Use this parameter to define the integral time, in units of 1 second. Integral action responds to the rate of change in the controller's CV output relative to the change in Error. The integral

		time you set is the amount of time, as calculated by the controller, required to bring the process to Set Point. Note that integral (wind up) error may be initialized via the Force Error Integral operation.
Td: Derivative Time	MI	Use this parameter to define the derivative time, in units of 1 second. Derivative action responds to the rate and direction of change in the Error. This means that a fast change in error causes a strong response from the controller. The derivative action 'anticipates' the PV's value in relation to the Set Point and adjusts the CV accordingly, thus shortening the PID function's response time.
ST: Sample Time	MI	Use this parameter to define the intervals between PID function updates, in units of 10mSecs.
Action: 0: Heat, 1: Cool	MB	Select Off to activate Reverse Action (control type = heating), ON to activate Direct Action (control type = cooling).
Input Range: Process Value Low limit	MI	Use this parameter to define the lower limit for the Process Value.
Input Range: Process Value High limit	MI	Use this parameter to define the upper limit for the Process Value.
Output Range: Control Value Low limit	MI	Use this parameter to define the lower limit for the Control Value.
Output Range: Control Value High limit	MI	Use this parameter to define the upper limit for the Control Value.

Parameters: **Outputs**	**Type**	**Function**
CV: Control Value	MI	CV is the output from the PID function. CV is output from the PID function and input to the process. Note that this output signal may be an analog or time-proportional variable value.
Status Messages Initialized to 0 when **Configuration** is activated.	MI	Value Message >=0 FB status OK < 0 -1 Proportion band zero. -2 Input range is invalid (PV input). -3 Output range is invalid (CV output). -4 Integral has reached maximum of 100,000. PID will not allow the Integral value to increase any further. -5 Error in Auto Tune vector addresses, ex., the vector exceeds the final address in the MI data type. -6 Set Point less then Input low range or Set Point more then Input high range. -7 to-10 Auto tune error. -11 Noise is more then 5% of Input Range.
Auto-tune parameters	MI	The start of a 32 MI-long Auto-tune vector that contains the Auto-tuned parameters.

Note • Note that, once you have run Auto-tune, you can back up the P, I, and D values, the sample time (ST), and the 32 MI-long Auto-tune vector into a Data Table. You can then transfer these values to another Vision controlling an identical system, in order to run PID without tuning the loop.

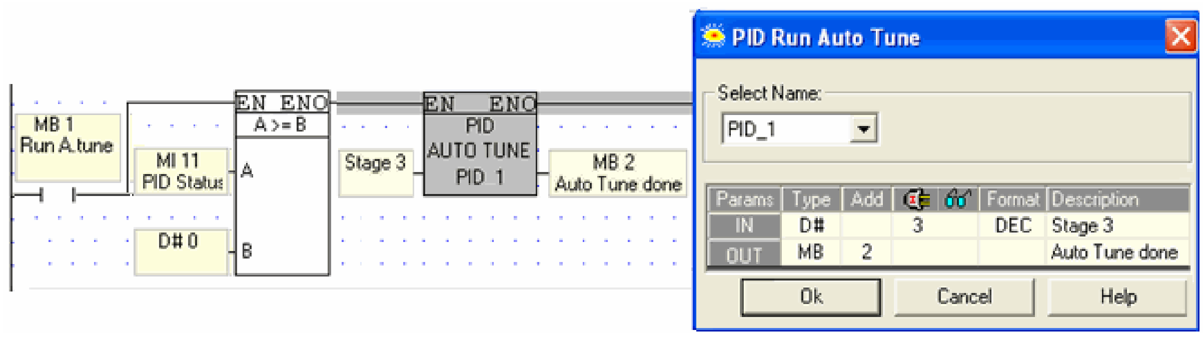
Run Auto-Tune

The Run Auto-tune operation uses the Configuration's parameters:

- Set Point
- Input Range:PV Low Limit & PV High Limit
- Output Range:CV Low Limit & CV High Limit

These values are used to Auto-tune the loop. After Auto-tune is run, the Auto-tune MB turns ON, and all of the Auto-tune parameters are written into the Autotune Parameter MI vector that is defined in the PID Configuration.

Note •	Note that, once you have run Auto-tune, you can back up the P, I, and D values, the sample time (ST), and the 32 MI-long Auto-tune vector into a Data Table. You can then transfer these values to another Vision controlling an identical system, in order to run PID without tuning the loop.
	In order to Auto-tune the loop, the PID Run operation must be suspended.



Select Name: PID_1

Params	Type	Add			Format	Description
IN	D#		3		DEC	Stage 3
OUT	MB	2				Auto Tune done

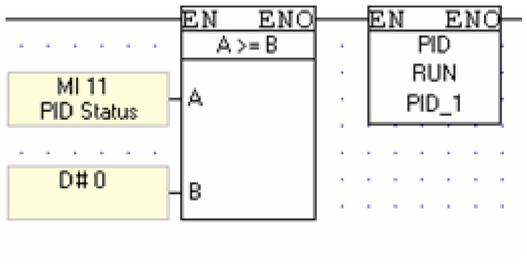
Auto-tune Parameters

Parameters: Inputs	**Type**	**Function**
Stage	#	The number of Stages aids the system to determine accurate Auto-tune parameters. The Default is 3. The higher the number of stages, the longer the Auto-tune time, however choosing a lower Stage may result in less accurate Auto-tune parameters.
Auto-tune Done	MB	After Auto-tune is run, the Auto-tune MB turns ON, and all of the Auto-tune parameters are written into the Auto-tune Parameter MI vector that is defined in the PID Configuration.

–

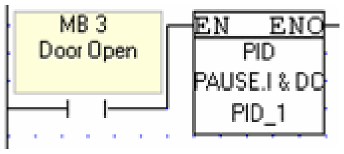
Run PID

In order to run a PID loop, the Run operation must be included in the application following the PID Configuration. In order to Auto-tune the loop, the PID Run must be suspended.



Pause Integral & Derivative Calculation

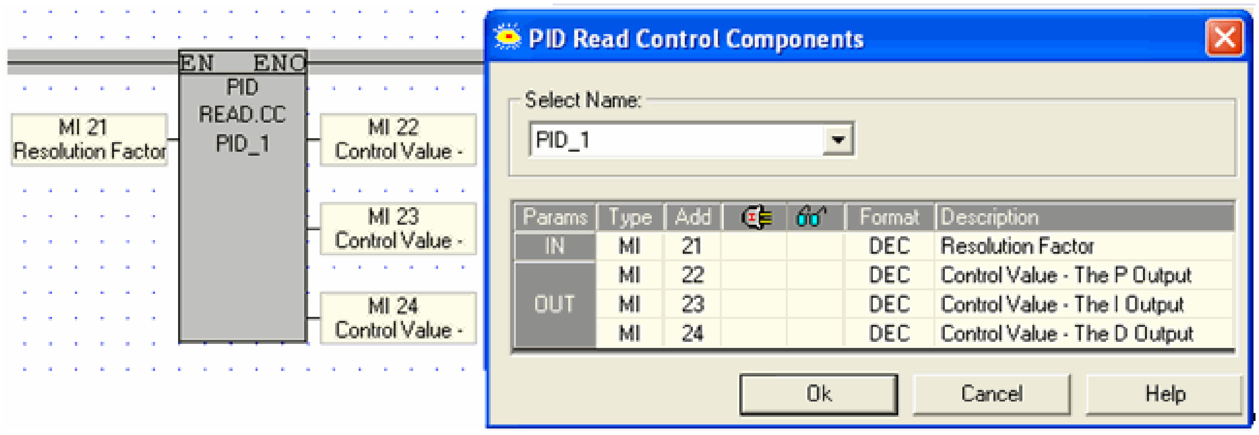
If conditions require, you can suspend this value and prevent it from changing. This may prove useful, for example, in a temperature application, when an opened oven door can cause a temporary temperature drop.



Read Control Components

This function enables you to scale down very large PID control values to smaller, more logical values. The current functions factors the PID control values by a value in an MI, then stores the values in the output MIs.

Use this function in conjunction with the PID Server utility to create a data file to send to support@unitronics in the event that you require technical support for PID applications.



Params	Type	Add			Format	Description
IN	MI	21			DEC	Resolution Factor
OUT	MI	22			DEC	Control Value - The P Output
	MI	23			DEC	Control Value - The I Output
	MI	24			DEC	Control Value - The D Output

Parameters	Type	Function
Resolution Factor	#	This is the value used to factor the PID control values.
Control Value: Proportional Output	MI	Stores the factored Proportional Output.
Control Value: Integral Output	MI	Stores the factored Integral Output.
Control Value: Derivative Output	MI	Stores the factored Derivative Output.

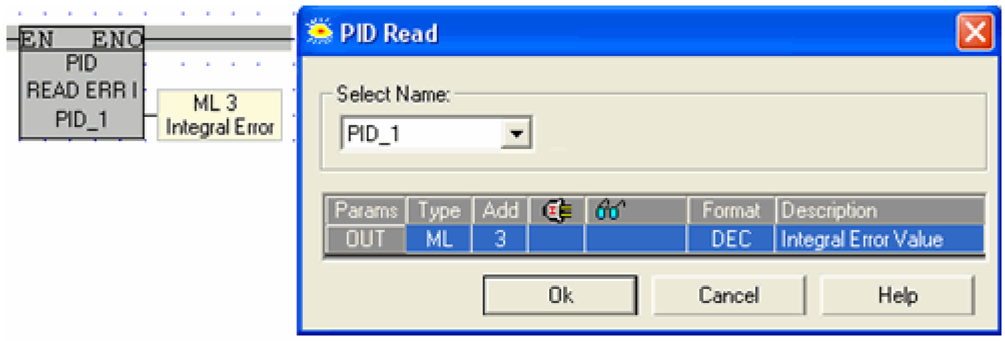
–

Error Integral

You can read and write to the Integral Value.

Read Error Integral

Use this operation to store the current error in the linked ML.

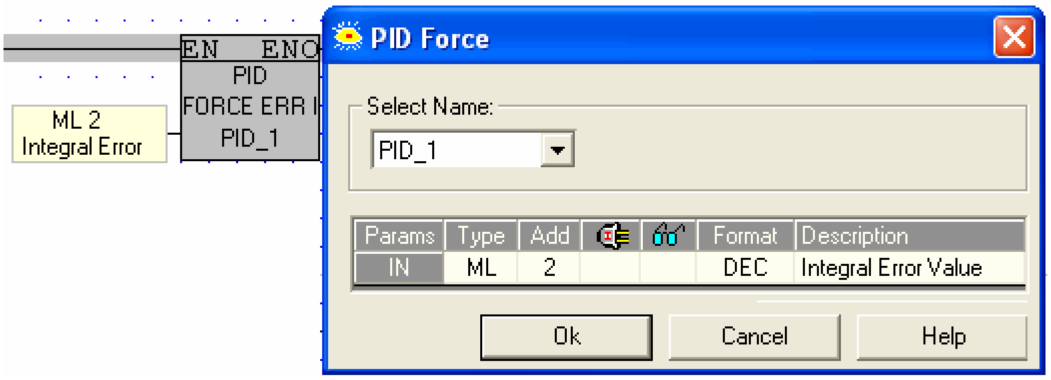


Select Name: PID_1

Params	Type	Add			Format	Description
OUT	ML	3			DEC	Integral Error Value

Force Error Integral

Use this to initialize or change the error value while the application is running. You can erase wind up error by writing '0' into the linked register.



Select Name: PID_1

Params	Type	Add			Format	Description
IN	ML	2			DEC	Integral Error Value

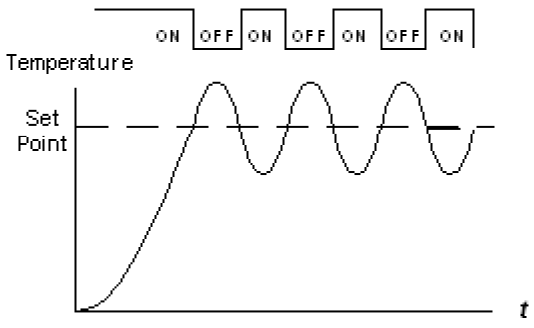
-

General Background: How PID Works

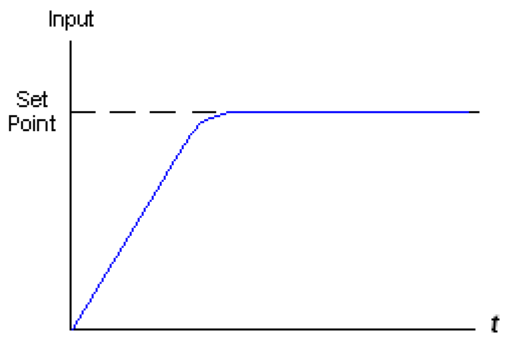
The PID function uses system feedback to continuously control a dynamic process. The purpose of PID control is to keep a process running as close as possible to a desired Set Point.

About PID and Process Control

A common type of control is On-Off control. Many heating systems work on this principle. The heater is off when the temperature is above the Set Point, and turns on when the temperature is below the Set Point. The lag in the system response time causes the temperature to overshoot and oscillate around the Set Point.



PID control enables you to minimize overshoot and damp the resulting oscillations.



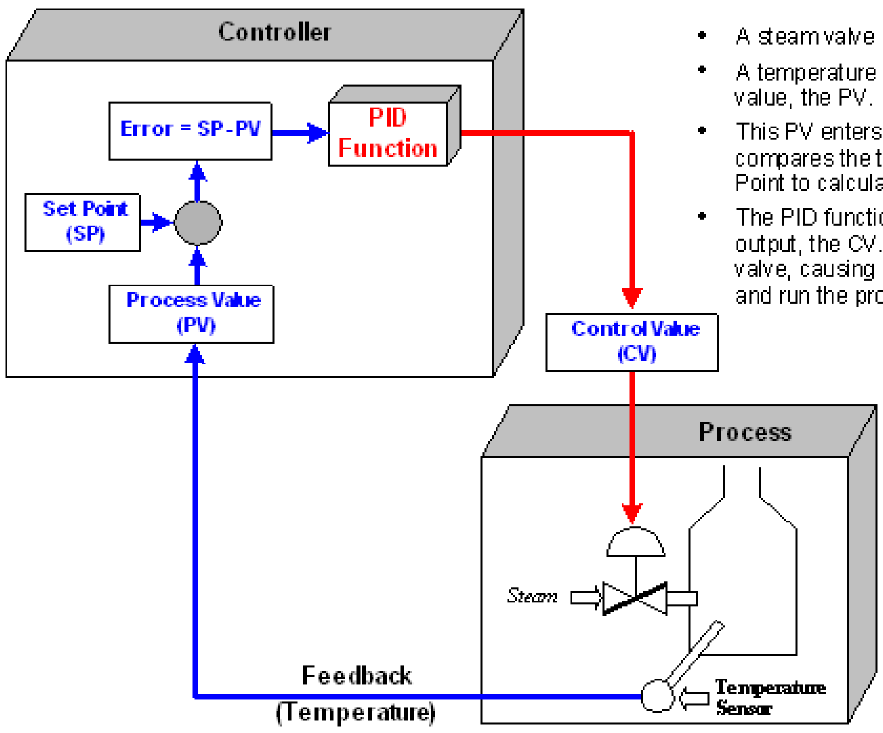
PID enables your controller to automatically regulate your process by:

Taking the output signal from the process, called the Process Variable (PV),

Comparing this output value with the process Set Point. The difference between the output Process Variable and the Set Point is called the Error signal.

Using the Error signal to regulate the controller output signal, called the Control Variable (CV), to keep the process running at the Set Point. Note that this output signal may be an analog or time-proportional variable value.

In the figure below, a system is regulated according to temperature.



- A steam valve releases heat into the system.
- A temperature sensor outputs a temperature value, the PV.
- This PV enters the controller. The PID function compares the temperature value with the Set Point to calculate the Error signal.
- The PID function regulates the controller output, the CV. This output controls the steam valve, causing it to vary the release of steam and run the process at the desired Set Point.

Inside the PID Function

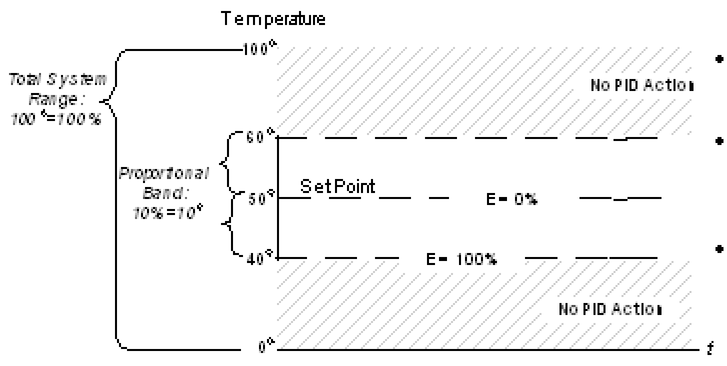
The PID function is based on 3 actions, Proportional, Integral, and Derivative. The PID output is the combined output of all 3 actions.

All of the PID functions are activated by changes in the process Error, the difference between the Process Value and the process Set Point value (E = SP – PV).

Proportional Band

The proportional band is a range defined around the Set Point. It is expressed as a percentage of the total Process Value (PV). When the PV is within this range, the PID function is active.

Note • The proportional band may exceed 100%. In this case, PID control is applied over the entire system range.



- The Process Value (PV) of this system is 0°-100°, a total range of 100°.
- The proportional band is set at 10 %. This means that the range of the proportional band is 40°-60°.
- When the Temperature is outside of the proportional band, the PID function is not active.

Proportional Action

Proportional action begins after the PV enters the proportional band; at this point, the Error is 100%. The action outputs a value that is in **direct linear proportion to the size of the Error value**.

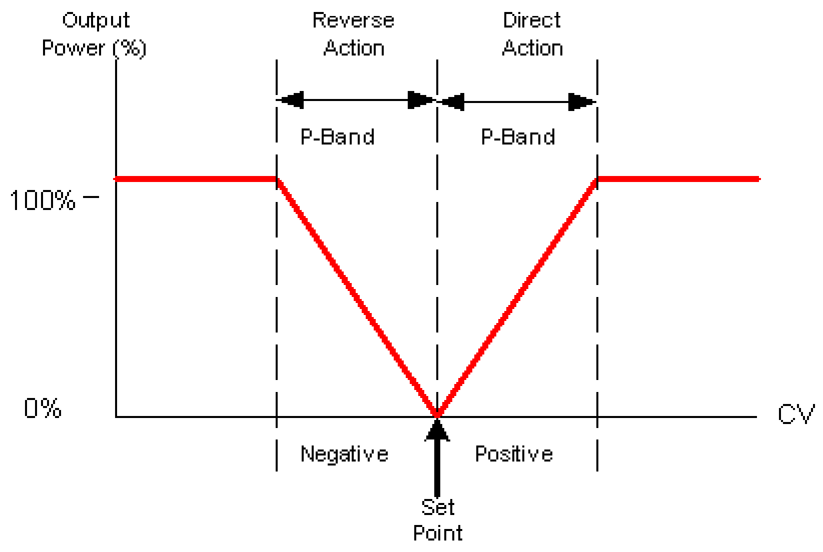
A broad proportional band causes a more gradual initial response from the controller. Typically, Set Point overshoot is low; but when the system stabilizes, oscillations around the Set Point tend to be greater.

A narrow band causes a rapid response that typically overshoots the Set Point by a greater margin. However, the system does tend to stabilize closer to the set point. Note that a proportional band set at 0.0% actually forces the controller into On-Off mode.

The drawback of proportional control is that it can cause the system to stabilize below set point. This occurs because when the system is at set point, Error is zero and the control value output is therefore pegged at zero as well. The majority of systems require continuous power to run at set point. This is achieved by integrating integral and derivative control into the system.

Direct and Reverse Action

Direct action causes the output to change in the same direction as the change in Error, meaning that a positive change in Error causes a positive change in the proportional band's output. Reverse action creates an inverse change in the output, meaning that a positive change in Error causes a negative change in output.

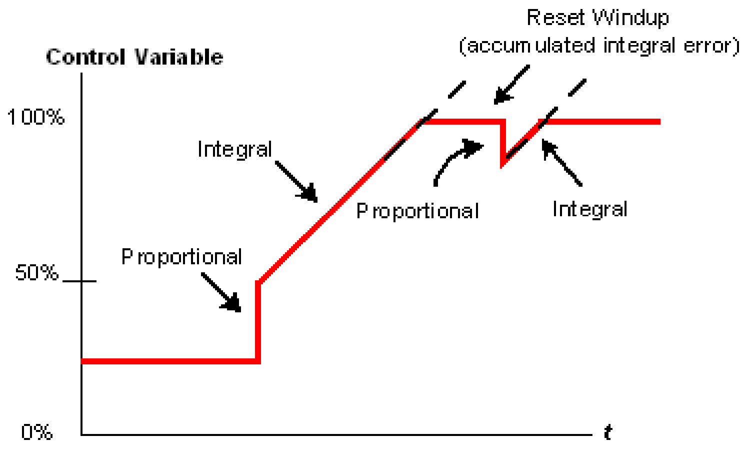


Integral Action

Integral action responds to the rate of change in the controller's CV output relative to the change in Error. The integral time you set is the amount of time, as calculated by the controller, required to bring the process to Set Point. Note that if you set a short integral time, the function will respond very quickly and may overshoot the Set Point. Setting a larger integral time value will cause a slower response. Integral time is sometimes called Reset.

The controller's CV output may reach and remain at 100%, a condition called saturation. This may occur, for example, if the process is unable to reach Set Point. This causes the Error signal to remain stuck in either the positive or negative range. In this situation, the integral action will grow larger and larger as the Error accumulates over time. This is called integral "wind up", which can cause the controller to overshoot the set point by a wide margin.

This situation can be prevented by setting an MB to clear the accumulated Integral error when saturation is occurs.



Derivative Action

Derivative action responds to the rate and direction of change in the Error. This means that a fast change in error causes a strong response from the controller.

The derivative action 'anticipates' the PV's value in relation to the Set Point and adjusts the controller's CV output accordingly, thus shortening the PID function's response time.

PID Status Messages

PID error indications are given in the Status Messages MI in the PID configuration.

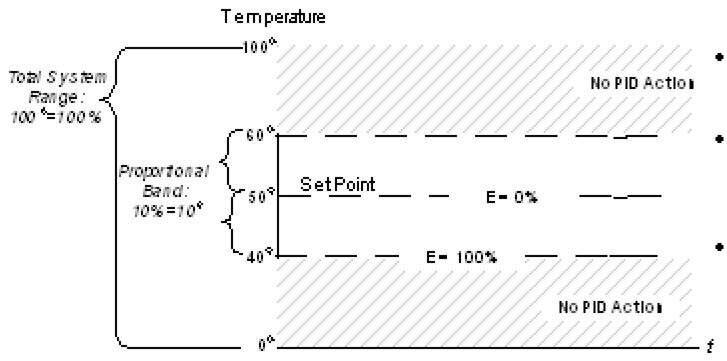
Status Messages Initialized to 0 when **Configuration** is activated.	MI	Value Message 0 FB status OK 1, 2, 3 Auto-tune in progress 4 PID running 5, 6 Setpoint change in progress 7 Integral-wind up 8 integral-wind down 9 Pause mode, Integral and Derivative values are not currently being calculated 10, 11 CV exceeds proportional band, no calculation performed 12, 13 AT parameter mismatch Note that this means that PID will run without Auto-tune. The user may either rewrite the PID values to the 32-MI long Auto-tune vector, or may re-run Auto-tune -1 Proportion band zero. -2 Input range is invalid (PV input). -3 Output range is invalid (CV output). -4 Integral Overflow has reached maximum of 100,000. PID will not allow the Integral value to increase any further. -5 Error in 32-MI long Auto Tune vector addresses, ex., the vector exceeds the final address in the MI data type. -6 Set Point less than Input low range or Set Point more than Input high range. -7 to-10 Auto tune error, failed to calculate PID parameters -11 Noise is more than 5% of Input Range.

PID Tips

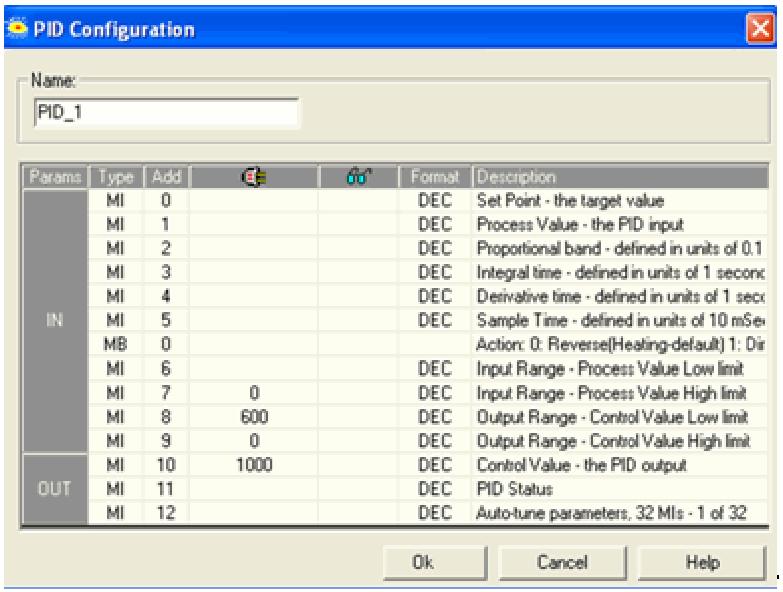
Background: the Proportional Band and PV

The proportional band is a range defined around the Set Point. It is expressed as a percentage of the total Process Value (PV). When the PV is within this range, the PID function is active.

Note •	The proportional band may exceed 100%. In this case, PID control is applied over the entire system range.
•	The PV and Setpoint must be the same unit type.


Temperature
100°
Total System Range: 100°=100%
60°
Proportional Band: 10%=10°
50°
SetPoint
E= 0%
40°
E= 100%
No PID Action
No PID Action
0°
t
• The Process Value (PV) of this system is 0°-100°, a total range of 100°.
• The proportional band is set at 10 %. This means that the range of the proportional band is 40°-60°.
• When the Temperature is outside of the proportional band, the PID function is not active.

You can set PV limits by assigning Power up values as shown below.



Name: PID_1

Params	Type	Add			Format	Description
IN	MI	0			DEC	Set Point - the target value
	MI	1			DEC	Process Value - the PID input
	MI	2			DEC	Proportional band - defined in units of 0.1
	MI	3			DEC	Integral time - defined in units of 1 seconc
	MI	4			DEC	Derivative time - defined in units of 1 seco
	MI	5			DEC	Sample Time - defined in units of 10 mSe
	MB	0				Action: 0: Reverse(Heating-default) 1: Dir
	MI	6			DEC	Input Range - Process Value Low limit
	MI	7	0		DEC	Input Range - Process Value High limit
	MI	8	600		DEC	Output Range - Control Value Low limit
	MI	9	0		DEC	Output Range - Control Value High limit
OUT	MI	10	1000		DEC	Control Value - the PID output
	MI	11			DEC	PID Status
	MI	12			DEC	Auto-tune parameters, 32 MIs - 1 of 32

Note •	A broad proportional band increases the stability of the system, but also increases fluctuations during the stable phase.
•	A proportional band that is too narrow will cause the system to react as though to ON-OFF control, and greatly overshoot and undershoot the setpoint.
TIP	You can increase the proportional band or the integral time to decrease overshooting and stabilize the system.

PID Analog Input Tips

PID Range Settings: PV Low and PV High limits.

If your PID PV is based on an analog input using current or voltage, you can use the analog range units to set the PV low and PV high limits. If this is done, you must also 'normalize' the setpoint to this range.

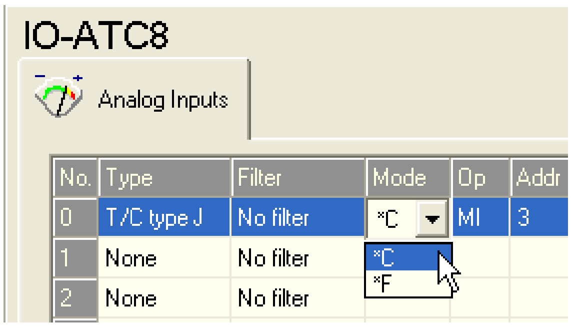
In general, it is recommended that you linearize the ranges to Engineering Units (EU). In this case use the analog reading ranges for X1 and X2, and use engineering units for Y1 and Y2.

Input resolution	Reading Range
10 bit, (0 to 10V, 0-20mA)	0-1023 units
10 bit (4-20mA)	205 to 1023
12 bit (0 to 10V, 0-20mA)	0-4095
12 bit (-20mA)	818-4095
14 bit (0 to 10V, 0-20mA)	0-16383 unit
14 bit (4-20mA)	3277-16383

Note • If you are using a PT100 or Thermocouple input, the values do not have to be linearized. This is because the input value is already in degrees Celsius or Fahrenheit (0.1 ° resolution) according to the Hardware Configuration.

Example: If the MI that is linked to a temperature input, set to Celsius, contains the value 385, the temperature reading is 38.5 ° C.

Example: Temperature, Thermocouple type 'J'

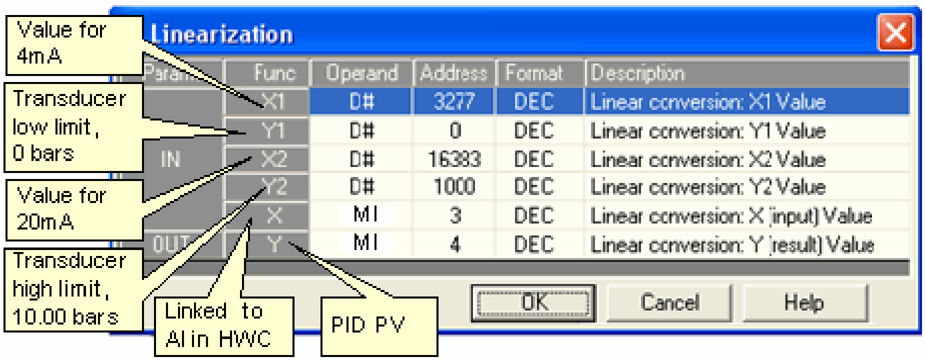

IO-ATC8

Analog Inputs

No.	Type	Filter	Mode	Op	Addr
0	T/C type J	No filter	°C	MI	3
1	None	No filter			
2	None	No filter			

Example: Linearizing to bars (pressure EU)

Assume that you are using a 14-bit input, 4-20mA with a pressure transducer with a range of 0-10 bars. Set Linearization parameters as shown below.



Param	Func	Operand	Address	Format	Description
IN	X1	D#	3277	DEC	Linear conversion: X1 Value
	Y1	D#	0	DEC	Linear conversion: Y1 Value
	X2	D#	16383	DEC	Linear conversion: X2 Value
	Y2	D#	1000	DEC	Linear conversion: Y2 Value
	X	MI	3	DEC	Linear conversion: X (input) Value
OUT	Y	MI	4	DEC	Linear conversion: Y (result) Value

Although you can set the PV limits to the input range, this may not produce accurate results.

Example 1: Assume that the 14-bit pressure transducer mentioned above is in a system with range limits of 0-5 bars. If you set the PV Low to 0 and the PV high to 1000 (10.00 bars), this is the range that will be used by the PID function. PID will work properly.

However, you can achieve better PID control by setting the PV Low to 0 and the PV High to 600 (6 bars). Since the error is a function of the PID working band, Example 2 will run with approximately 40% greater accuracy than Example 1.

Example 2: Thermocouple type 'J' has as range of -200 to 760 ° C. Assume that this is the input for a PID system with an ambient temperature that can reach a low of 10 ° C to a maximum setpoint of 250 ° C. If you set the PV Low to 0 and the PV High to 300 ° C, PID control will function with approximately 3 times greater accuracy than if you set the PV range to cover the entire -200 to 760 input range.

PID via Digital Output (Contactor, Solenoid Valve, SSR)

You can use the PWM (Pulse Width Modulation) FB to control a PID system. PWM FB enables you to control the ratio between the ON and OFF status of a selected MB (Duty Cycle) within the defined cycle time, which is given as ticks of 2.5 milliseconds. The ratio is given as an ON pulse percentage on the range of 0-1000 (0-100%).

In order to control PID with the PWM FB, you **must** set the PID CV range to 0-1000 (0-100.0%).

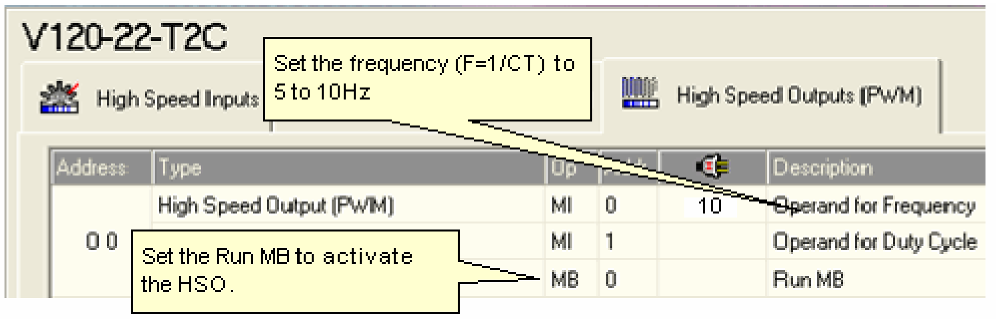
Since the PWM cycle time is set in ticks of 2.5 ms, e.g. 1 s = 400. If the output is a relay/contactor/solenoid valve, the recommended cycle time range is between 2000 and 12000 (5 to 30 sec).

Example: As the PWM output MB pulses ON and OFF, the pulse ON time is proportional to the CV. If the cycle time is = 4000 (10 sec), and the CV is 100 (10.0%), the output bit will be ON for 1 second and OFF for 9 seconds, thereby supplying 10% of the energy to the system.

If the switch is an SSR, the recommended cycle time range is between 200 and 1000 (0.5 to 2.5 sec). Transistor outputs are preferable.

If you use an SSR, you can use a Unitronics PLC that supports a high-speed output. In Hardware Configuration, set the Frequency (F=1/CT) parameter's

Power-up Value to 5 to 10Hz (which is 0.2 -0.1s cycle time). Link the PWM Duty Cycle MI to the CV. Note that in order to activate the output pulses, you must SET the RUN MB linked to the HSO in Hardware Configuration.


V120-22-T2C
High Speed Inputs
High Speed Outputs (PWM)
Set the frequency (F=1/CT) to 5 to 10Hz
Set the Run MB to activate the HSO.

Address	Type	Op	Addr		Description
	High Speed Output (PWM)	MI	0	10	Operand for Frequency
0 0		MI	1		Operand for Duty Cycle
		MB	0		Run MB

Manual Loop Tuning

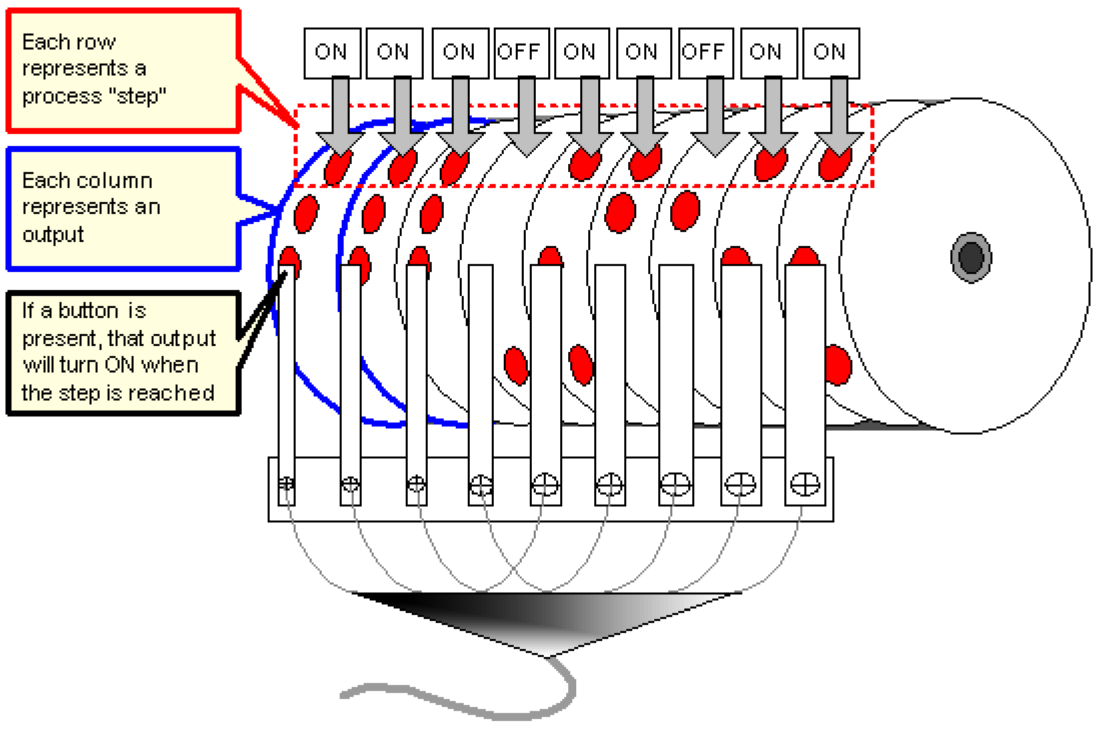
In certain cases, you may already have the PID values required by you application. However, you should still perform Autotune in order to ensure proper PID function.

This is because during the Autotune process, the PID function collects certain essential data. Unitronics' proprietary PID algorithm uses this data to run smooth, accurate PID.

Drum Sequencer

Drum Sequencer Overview

The Drum Sequencer FB simulates a mechanical drum sequencer as shown below. Drum instructions are best suited for repetitive processes that consist of a finite number of steps.


Each row represents a process "step"
Each column represents an output
If a button is present, that output will turn ON when the step is reached
ON
ON
ON
OFF
ON
ON
OFF
ON
ON

The Drum Sequencer FBs enable you to:

- Define the number of steps on your 'drum'
- Define the time duration of a step
- Define the number of output columns
- Link a coil (O or MB) to an output column.
- In each step, determine the status of each coil.

The Drum FBs are located on the FB's menu.

Using Drum

The main routine of a basic drum Ladder application must include a:

- Configuration
  The Configuration contains the Drum Outputs, arranged in steps. The duration time of each step is also set in the configuration.
- Scan
  This should follow the Configuration. The Scan controls the drum; if the Scan is inactive, the drum is static.

These elements enable the drum to begin at a Start Step, which is specified in the Configuration. When the duration time set for the first step elapses, the drum progresses to the next step. When the last step in the drum is complete, the drum continues with the Start Step.

Progressing through Steps

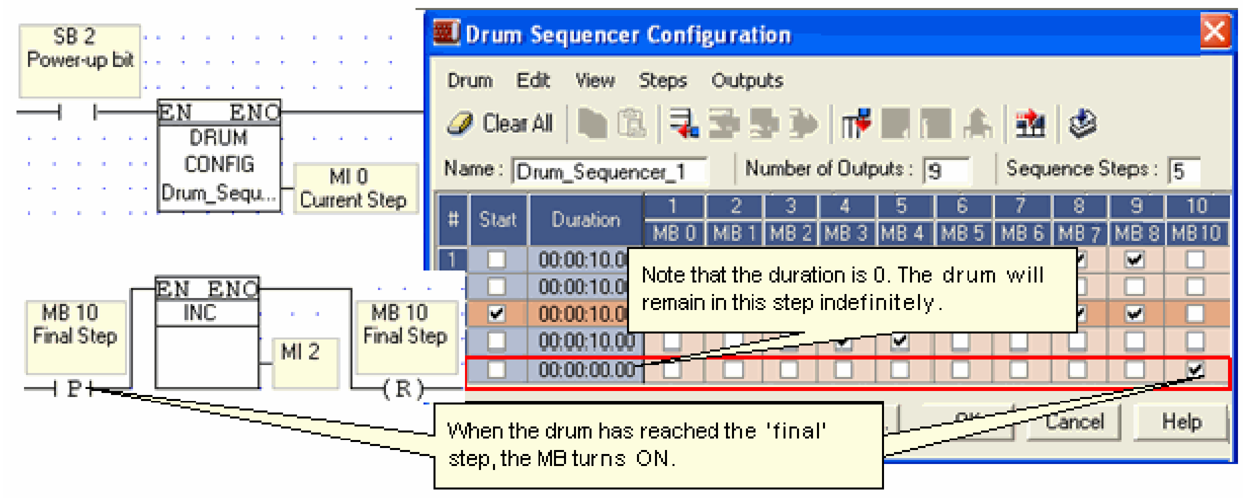
- Set a duration time
  This is done in the Configuration. If you do **not** set a time, the drum will **remain** in that step, unless you use a Jump to Step operation.
- Jump to another step at any time
  Even if the drum is in the middle of a different step, you can use a Jump to Step operation: Next Step, Go To Start Step, or Go To Step (Index)

In the event of Power-Up

At Power-up, the drum starts at the Start Step. If power-up occurs while the drum is running, you can return to the desired step by saving the Current Step Index, and then using that step number to as a Power-up value in a Go To Step Index operation.

Final Step

In order to mark the final step in the drum, use a step that includes an output that you dedicate to that purpose. You can then use the changing output status to drive any task.



Name : Drum_Sequencer_1
Number of Outputs : 9
Sequence Steps : 5

Note that the duration is 0. The drum will remain in this step indefinitely.

When the drum has reached the 'final' step, the MB turns ON.

Drum Sequencer Operations

Configuration

Scan

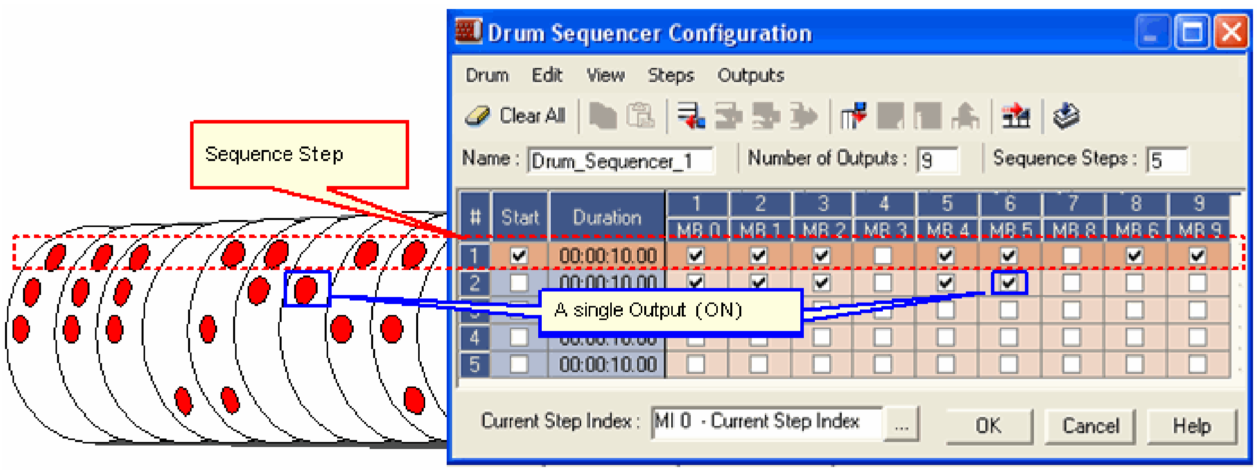
Go To Step

Go To Start Step

Go To Next Step

Drum Sequencer: Configuration

The Drum Sequencer Configuration represents the traditional 'drum''. The Configuration contains rows of output bits. Each row is a Sequence Step, which correspond to the 'process steps' on a drum.

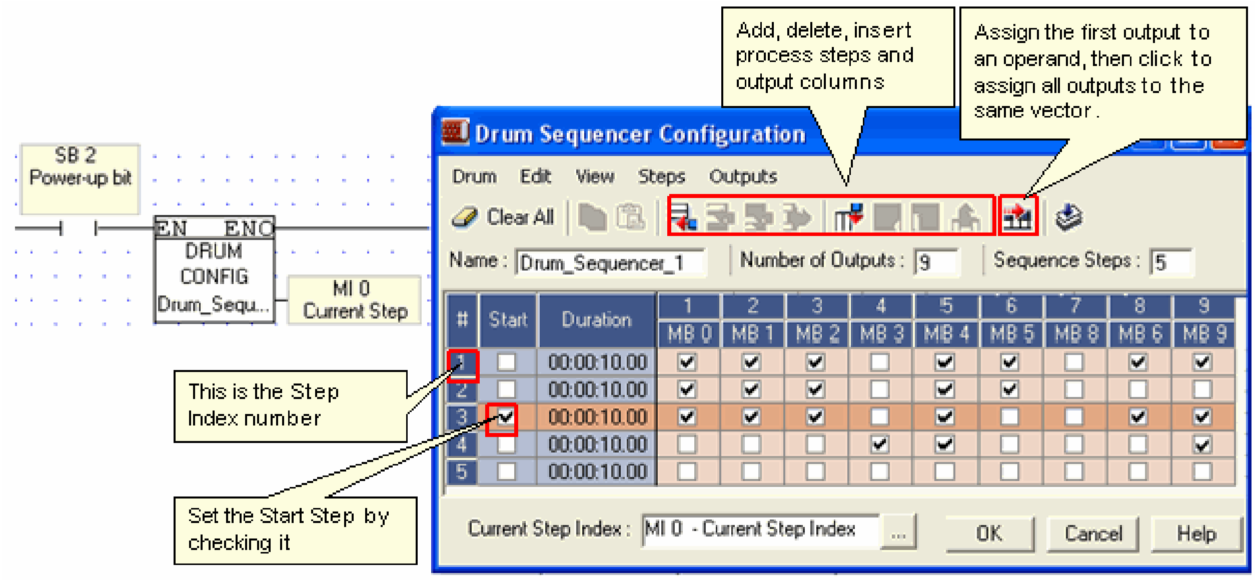


Name : Drum_Sequencer_1 | Number of Outputs : 9 | Sequence Steps : 5

#	Start	Duration	1 MB 0	2 MB 1	3 MB 2	4 MB 3	5 MB 4	6 MB 5	7 MB 8	8 MB 6	9 MB 9
1	✓	00:00:10.00	✓	✓	✓		✓	✓		✓	✓
2		00:00:10.00	✓	✓	✓		✓	✓			
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]							
4		[illegible]									
5		00:00:10.00									

Current Step Index : MI 0 - Current Step Index

Your program must activate the Drum Sequencer Configuration before running Drum Sequence operations. You can do this by activating it as a power-up task.


Add, delete, insert process steps and output columns
Assign the first output to an operand, then click to assign all outputs to the same vector.
SB 2
Power-up bit
EN
ENO
DRUM
CONFIG
Drum_Sequ...
MI 0
Current Step
This is the Step Index number
Set the Start Step by checking it
Drum Sequencer Configuration
Drum Edit View Steps Outputs
Clear All

Name : Drum_Sequencer_1 | Number of Outputs : 9 | Sequence Steps : 5

#	Start	Duration	1 MB 0	2 MB 1	3 MB 2	4 MB 3	5 MB 4	6 MB 5	7 MB 8	8 MB 6	9 MB 9
1		00:00:10.00	✓	✓	✓		✓	✓		✓	✓
2		00:00:10.00	✓	✓	✓		✓	✓			
3	✓	00:00:10.00	✓	✓	✓		✓			✓	✓
4		00:00:10.00				✓	✓				✓
5		00:00:10.00									

Current Step Index : MI 0 - Current Step Index

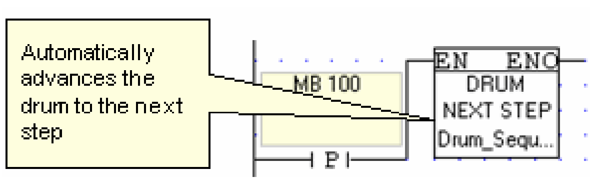
Name	Description
#	This the Step Index number. Use this together with the Go To Step Index operation, which you use for determining the Drum operations' functions.
Start	When the Drum Sequencer runs, this is the first step in the sequence. Whenever the drum restarts, whether as a result of power failure or any other reason,
Duration	If you do **not** set a Time, the drum will **remain** in that step, unless a Jump to Step operation changes the current step.
Current Step Index	This MI contains the Step Index number of the current drum process step.

Drum Sequencer: Jump to Step

To jump to another step at any time, even if the drum is in the middle of a different step, link one of these operations to a Configuration.

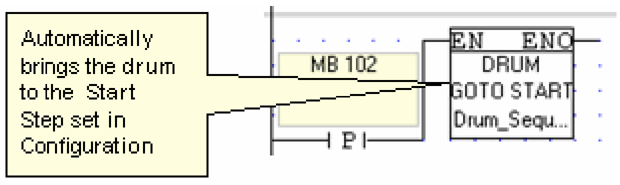
Next Step

Use this to immediately progress to the next step, without relation to the current state of the drum.



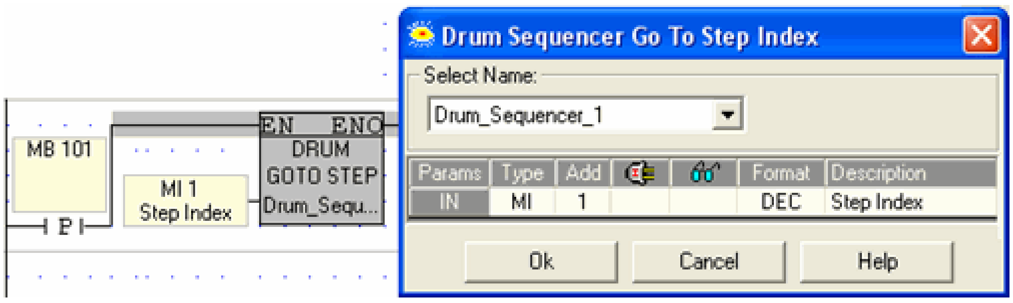
Go To Start Step

Use this to jump back to the Start Step set in the Configuration..


Automatically brings the drum to the Start Step set in Configuration
MB 102
P
EN ENO
DRUM
GOTO START
Drum_Sequ...

Go To Step (Index)

Use this to immediately jump to a particular step, according to its Step Index number.



Select Name: Drum_Sequencer_1

Params	Type	Add			Format	Description
IN	MI	1			DEC	Step Index

In the event of Power-Up

At Power-up, the drum starts at the Start Step. If power-up occurs while the drum is running, you can return to the desired step by saving the Current Step Index, and then using that step number to as a Power-up value in a Go To Step Index operation.

Events

Events Overview

Events: Registering MB status change

An Event is the change in status of an MB from OFF (0) to ON (1). Events can be used, for example, to monitor the status of an array of alarm bits. The Events FBs are located on the FB's menu.

The Event: Scan enables you to:

- Define a vector of MBs.
- Locate the **first** MB that is ON (active) within that vector
- Record the MB's location.

Other Event operations enable you to:

- Move between the active MBs within that vector
- Change MB status from ON to OFF.

Note • From OS Version OS 1.3.00 and up, V130/V350/V570 support XBs.

Event Operations

Event: Scan

Next Event

Previous Event

Clear Current

Clear All

Events: Count

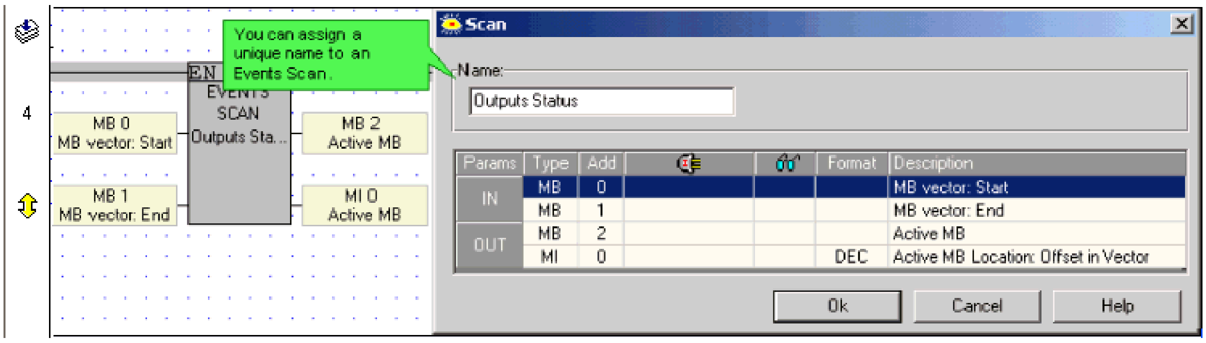
Events: Loopback

Track Events

Events: Scan

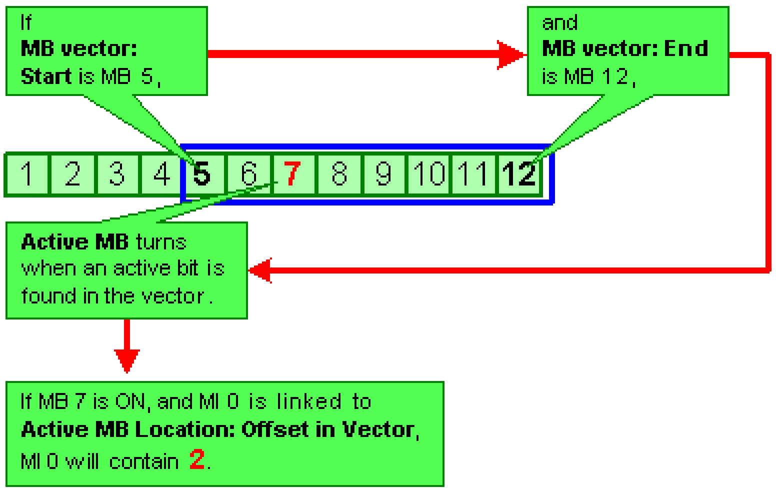
Use the Events: Scan to define a vector of MBs and locate the first positive bit within that vector. Once you have defined a vector of MBs using the Events Scan FB, you can perform other actions within that vector using Event operations.

Note • Place the Scan FB directly on the left-hand Ladder rail; this ensures that all of the Events functions will run continuously.



Name:
Outputs Status

Params	Type	Add			Format	Description
IN	MB	0				MB vector: Start
	MB	1				MB vector: End
OUT	MB	2				Active MB
	MI	0			DEC	Active MB Location: Offset in Vector

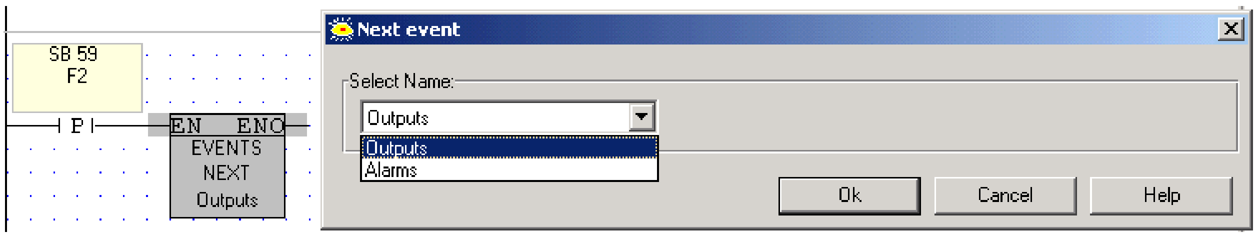

If **MB vector: Start** is MB 5,
and **MB vector: End** is MB 12,
1 2 3 4 5 6 7 8 9 10 11 12
Active MB turns when an active bit is found in the vector.
If MB 7 is ON, and MI 0 is linked to **Active MB Location: Offset in Vector**, MI 0 will contain 2.

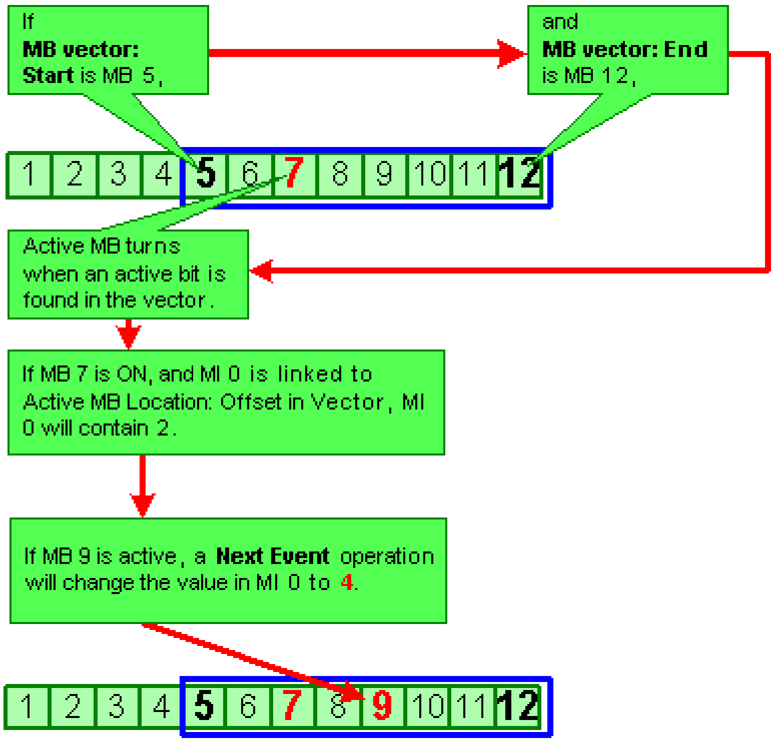
Operand	**Type**	**Function**
MB Vector: Start	MB	The first MB of the vector to be scanned.
MB Vector: End	MB	The last MB of the vector to be scanned.
Active MB	MB	Turns ON when an active bit is located. **Note** • Once the Active MB has been turned ON, it remains ON until it is reset by the program.
Active MB Location: Offset in Vector	MI, ML,DW	Contains the location of the Active MB, relative to the beginning of the defined vector. **Note** • If no MB is active, the value in the linked register will be -1.

Next Event

Once a vector of MBs is defined using the Event Scan FB, the Scan finds the first active MB in the vector.

Use the Next Event operation to move to the next active MB in the vector, in the direction of the Most Significant Bit.

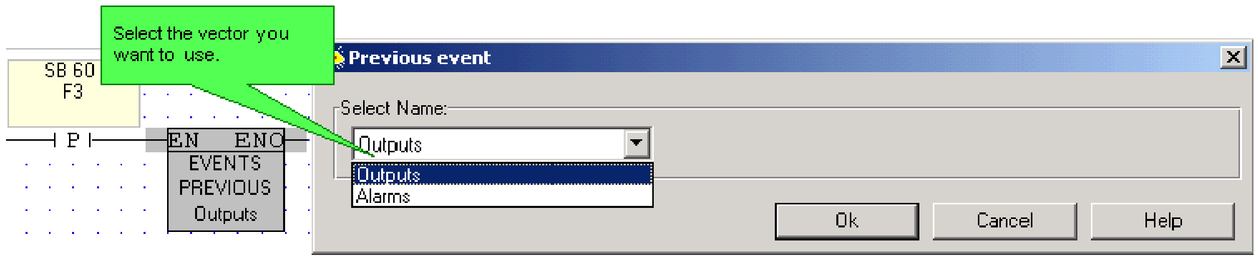


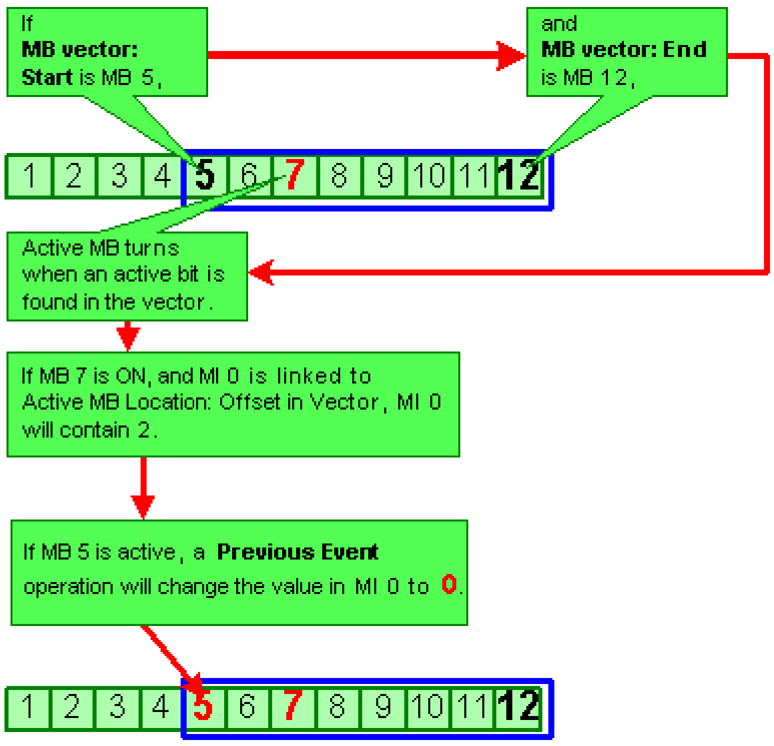

If **MB vector: Start** is MB 5,
and **MB vector: End** is MB 12,
1 2 3 4 5 6 7 8 9 10 11 12
Active MB turns when an active bit is found in the vector.
If MB 7 is ON, and MI 0 is linked to Active MB Location: Offset in Vector, MI 0 will contain 2.
If MB 9 is active, a **Next Event** operation will change the value in MI 0 to 4.
1 2 3 4 5 6 7 8 9 10 11 12

Previous Event

Once a vector of MBs has been defined using the Event Scan FB, the Scan finds the first active MB in the vector.

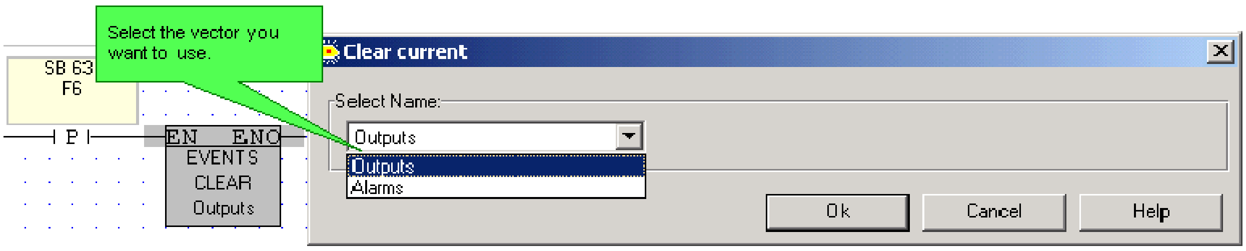
Use the Previous Event operation to move to the previously active MB in the vector, in the direction of the Least Significant Bit.

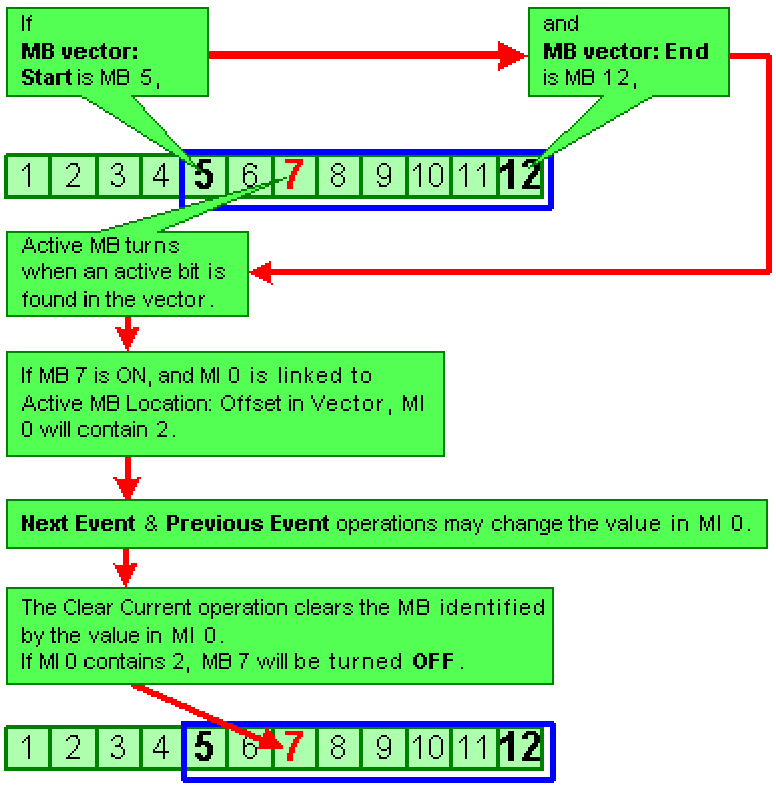




Clear Current

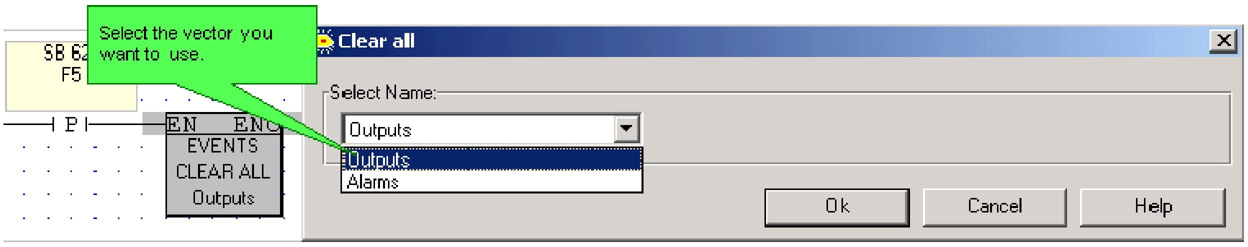
Use the Clear Current operation to reset a currently active MB.




If
MB vector: Start is MB 5,
and
MB vector: End is MB 12,
1 2 3 4 5 6 7 8 9 10 11 12
Active MB turns when an active bit is found in the vector.
If MB 7 is ON, and MI 0 is linked to Active MB Location: Offset in Vector, MI 0 will contain 2.
Next Event & Previous Event operations may change the value in MI 0.
The Clear Current operation clears the MB identified by the value in MI 0.
If MI 0 contains 2, MB 7 will be turned OFF.
1 2 3 4 5 6 7 8 9 10 11 12

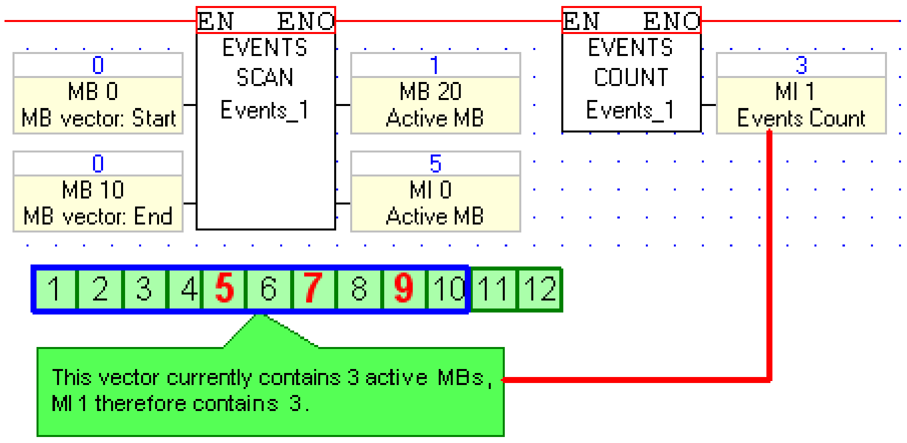
Clear All

This operation causes **all** of the active MBs in an Event vector to be reset.



Events: Count

This shows the number of MBs currently ON within the vector defined by the Events Scan.



Events: Loopback

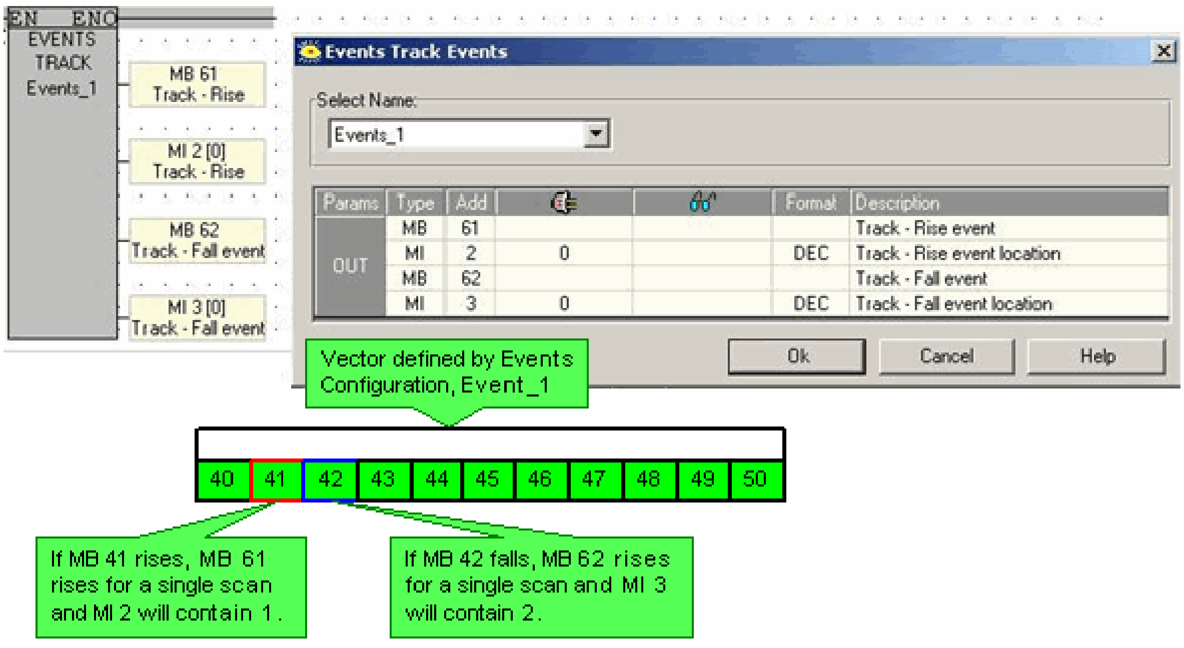
When Loopback is activated by power flow, the focus of the Events moves to the beginning of the vector.

Track Events

This FB enables you to keep track of Rise and Fall events as they happen.

- Rise Events:
  Each time an MB rises in the Events vector, the Rise Event MB in Track Events turns ON for a single cycle, and the Rise Location MI contains the location of the active MB, relative to the beginning of the Events vector. If two MBs rise during a single cycle, the Rise Event Location MI will show the events in two consecutive cycles.
- Fall Events:
  Each time an MB falls in the Events vector, the Fall Event MB in Track Events turns ON for a single cycle, and the Fall Location MI contains the location of the active MB, relative to the beginning of the Events vector. If two MBs fall during a single cycle, the Rise Event Location MI will show the events in two consecutive cycles.

Note that Track events can only monitor an vector that, as defined in the Events Configuration, does not exceed 255 MBs in length.



Params	Type	Add			Format	Description
OUT	MB	61				Track - Rise event
	MI	2	0		DEC	Track - Rise event location
	MB	62				Track - Fall event
	MI	3	0		DEC	Track - Fall event location

Vector defined by Events Configuration, Event_1

If MB 41 rises, MB 61 rises for a single scan and MI 2 will contain 1.

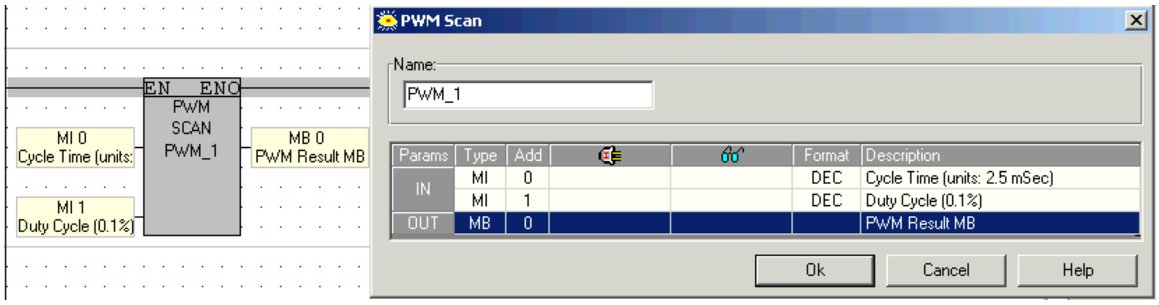
If MB 42 falls, MB 62 rises for a single scan and MI 3 will contain 2.

PWM

PWM FB Overview

The PWM FB enables you to control the change in the status of a selected MB, causing it to function like a PWM (Pulse Width Modulation) output at a resolution of 2.5 milliseconds (in relation to the total scan time).

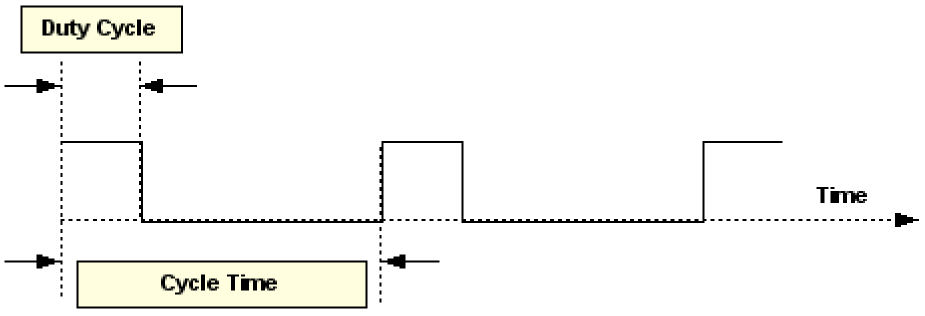
You can place the PWM FB directly on the left Ladder rail to cause it to run continuously. PWM is located on the FBs menu.



Name: PWM_1

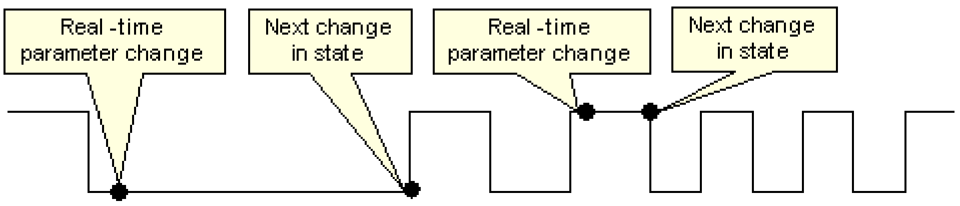
Params	Type	Add			Format	Description
IN	MI	0			DEC	Cycle Time (units: 2.5 mSec)
	MI	1			DEC	Duty Cycle (0.1%)
OUT	MB	0				PWM Result MB

Parameter	Function
Cycle Time	The total cycle time. One unit of cycle time is equal to 2.5 msec.
Duty Cycle	The ratio of the "on" period of a cycle to the total cycle period, expressed as a tenth of a percent of the total Cycle Time.
Output	The MB you select will function as the PWM output. It will be turned ON and OFF according to the input parameters.



Real-time Cycle Changes

If you link the parameters Cycle Time or Duty Cycle to MIs, and the parameters change when the application is running, note that the change is effective at the **next** change in status.



Example

Cycle Time = 800

Duty Cycle = 255

If the Cycle Time parameter is 800; the **total** PWM cycle time will be 2000mSec (800 x 2.5 = 2000mSec).

If the Duty Cycle parameter is 255; the selected MB will be ON for **25.5% of the total Cycle Time** (25.5% of 2000 = 510mSec), and OFF for **74.5% of the total Cycle Time** (74.5 of 2000= 1490mSec)

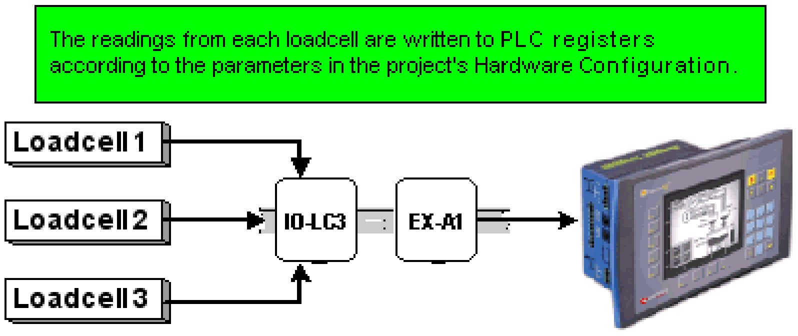
Loadcell

Loadcell Overview

Loadcell FBs, located on the **FB** menu, enable you to include an I/O module that is connected to a loadcell or strain-gauge in your control application. Unitronics I/O expansion loadcell modules are intelligent I/O modules that are capable of receiving analog values directly from loadcells.

IO-LC1 offers 1 Loadcell input; IO-LC3 module offers 3 Loadcell inputs. Each IO-LCx module is capable of providing excitation for up to 12 loadcells.

The **Loadcell FB**s enable you to calibrate the loadcell. You can also tare and zero the loadcell, compensate for deadload and scale movement, and set the input range.


The readings from each loadcell are written to PLC registers according to the parameters in the project's Hardware Configuration.
Loadcell 1
Loadcell 2
Loadcell 3
IO-LC3
EX-A1

Once you connect the loadcell and calibrate at least 2 points, you can begin to run a loadcell application. The loadcell input can be read in 6 different ways:

- Gross weight
- Net weight
- Net Min. Weight
- Net Max. Weight
- Scaled to uV/V
- Raw Value

Most applications will require only the Gross or Net weight. Raw Value and uV/V readings may be useful for troubleshooting purposes.

Note •	Minimum settling times for projects using multiple loadcells are • 12.5ms for one active loadcell • 675ms for two active loadcells • 1,012.5ms for three active loadcells More information is available in the Setup Help topic, and in the module's specification sheet..
•	Both negative and positive (signed and unsigned) values can be processed by the I/O-LCx and the support software, enabling a range of applications.
•	This feature is not supported by the V120-12 series.

FB Operations

Configuration

Scan

Calibration

Setup

Tare & Zero

Advanced

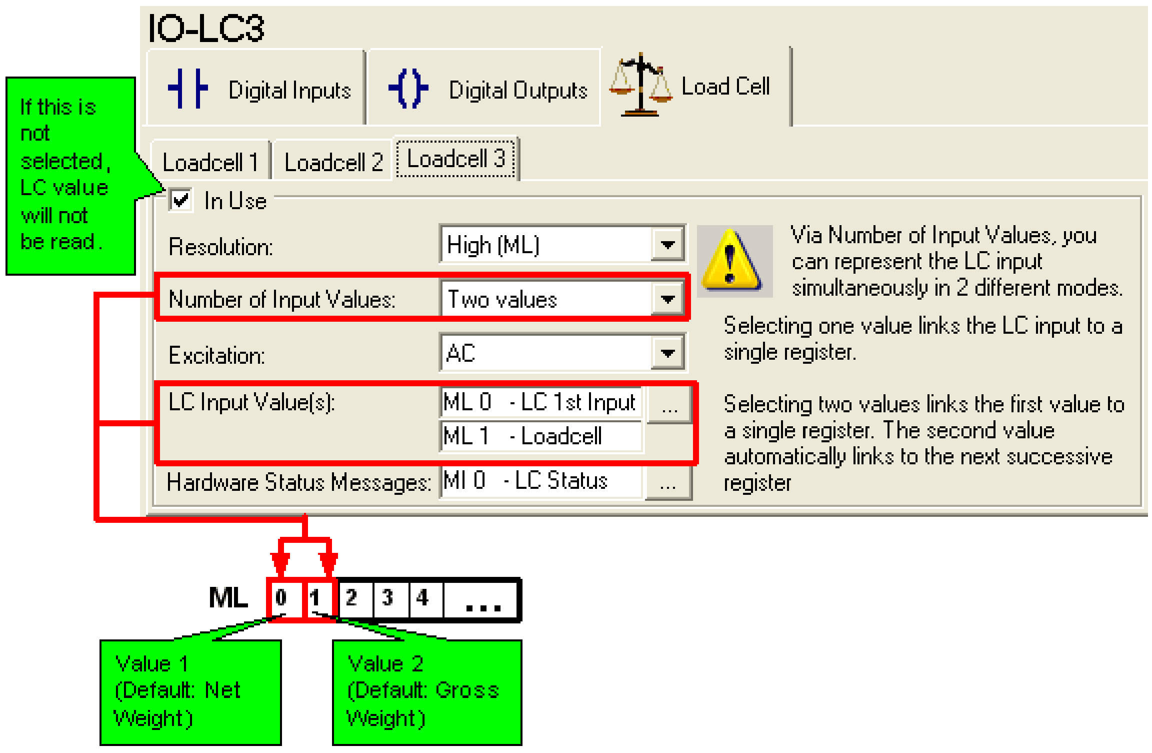
Note •	Both negative and positive (signed and unsigned) values can be processed by the I/O-LCx and the support software, enabling a range of applications.
•	This feature is not supported by the V120-12 series.

Loadcell Hardware Configuration

The IO-LCx Hardware Configuration enables you to configure a loadcell, plus the digital input and digital outputs located on the module. These digital I/Os enable you to implement setpoints that are processed within the I/O module, independently of the controller and its program scan, enabling a fast response to process events.

Configuring a Loadcell

The number of Loadcell tabs in the Hardware Configuration window depends upon the loadcell module.


IO-LC3
Digital Inputs
Digital Outputs
Load Cell
Loadcell 1
Loadcell 2
Loadcell 3
If this is not selected, LC value will not be read.
ML
0
1
2
3
4
...
Value 1 (Default: Net Weight)
Value 2 (Default: Gross Weight)

☑ In Use

Resolution: High (ML)

Number of Input Values: Two values

Excitation: AC

LC Input Value(s): ML 0 - LC 1st Input; ML 1 - Loadcell

Hardware Status Messages: MI 0 - LC Status

Via Number of Input Values, you can represent the LC input simultaneously in 2 different modes.

Selecting one value links the LC input to a single register.

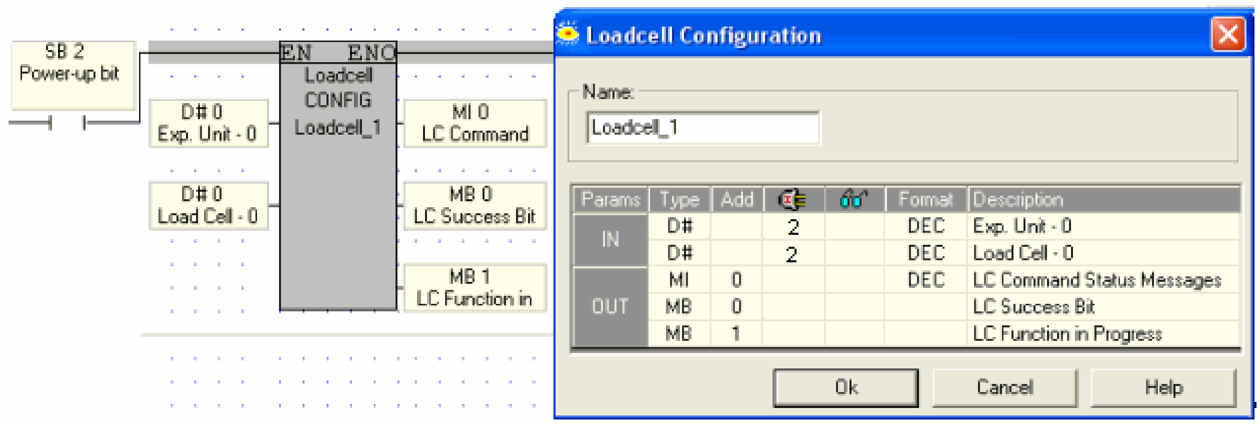
Selecting two values links the first value to a single register. The second value automatically links to the next successive register

Parameter	Type	Function
In use		Select 'In Use' to enable the loadcell for the application. **Note** • A loadcell marked 'in use' can be suspended according to application conditions via the Advanced Calibration function Enable/Disable loadcell. This may be done to shorten the application's calibration time.
Resolution	ML, MI	Selecting High enables you to link the input value to an ML, Normal to an MI.
LC Input Value(s)	MI, ML	When the application runs, these registers contain the weight value input to the controller from the I/O LCx. When you select two values, you link the first value to a single register. The second value is **automatically linked** to the next register. The default representation mode for the first register is **Net Weight**, and **Gross Weight** for the second register.
Excitation		AC is the default, recommended Excitation method. You may select the DC option if your application requires.
Hardware Status Messages	MI	Provides a bitmap showing the status of the module.

Loadcell: Configuration

All Loadcell operations run through a Loadcell Configuration placed in the control program. Each Configuration is linked to a specific Loadcell input on the I/O expansion module.

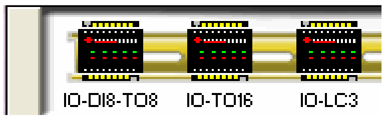
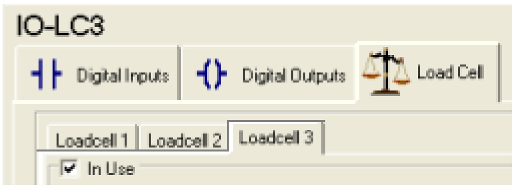
Loadcell Configuration is generally a Power-up task. The Loadcell Configuration FB is located on the FB's>Loadcell menu.



Name: Loadcell_1

Params	Type	Add			Format	Description
IN	D#		2		DEC	Exp. Unit - 0
	D#		2		DEC	Load Cell - 0
OUT	MI	0			DEC	LC Command Status Messages
	MB	0				LC Success Bit
	MB	1				LC Function in Progress

Parameter	Type	Function
I/O Module DIN Location		Links the Configuration with the correct I/O expansion module according to its DIN rail location in Hardware Configuration.

		On this DIN rail, the I/O LC3 is module # 2  **Note •** If the module is not located in the entered location, the LC Command Status Messages MI will display 6, Communication Error (I/O module does not exist)
Loadcell Input Number		Here, Loadcell 2 is being shown as marked 'In Use'.  IO-LC3 Digital Inputs Digital Outputs Load Cell Loadcell 1 Loadcell 2 Loadcell 3 In Use **Note •** If the selected Loadcell is not marked 'In Use', the LC Command Status Messages MI will display 11, Illegal parameter
LC Command Status Messages	MI	Is reset to 0 when a command is activated. Updated at the end of all operations using the Loadcell Configuration. Indicates error status for all loadcell operations processed by the Configuration Current value always shows the most **recent** error status. Value Message 0 Function in Progress 1 Command carried out successfully 2 I/O Expansion Command Buffer is full, please retry. Can be avoided by using SB 91, I/O Expansion Module--Command Buffer Full, as a condition 3 The I/O expansion module linked to the configuration is busy 5 Timeout Exceeded 6 Communication Error (I/O module does not exist) 11 Illegal parameter 13 Power supply not connected 16 Scale is currently in motion (is only relevant if In-Motion function is applied) 17 Signal is out of range (this value occurs when the Out of Range bit is ON) 18 Illegal weight (Occurs during calibration, if the raw value of weight being calibrated is too close to the raw value of an already calibrated weight; minimum distance is 256 or 100 Hex) 19 Command not supported in uV/V mode 20 Not calibrated (This value appears when less than 2 points have been calibrated) 21 EEPROM Protection Error (Indicates when too many Save Calibration FBs are run too frequently. Check the activating conditions for the Save Calibration FB, and whether your application contains loops)
LC Success Bit	MB	Use this to monitor the status of a command Turns ON when: Configuration is first activated. Command is successfully processed. Command Status Messages MI is 1. **Turns OFF when:** Command Status Messages MI is 0. Function in Progress MB is ON, indicating that a command is being processed. Command error occurs

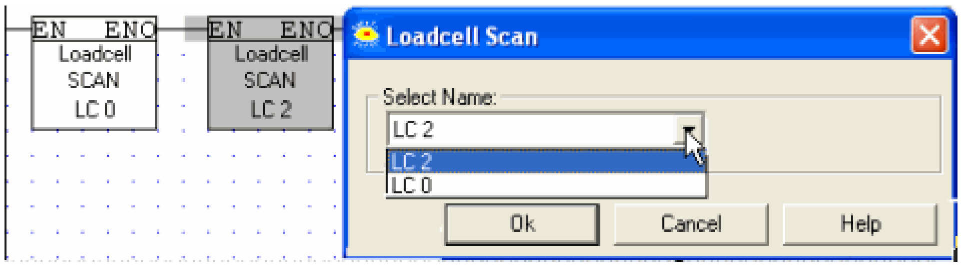
Function in Progress	MB	This bit is ON when the module is processing a command. Use this as a condition bit for Loadcell operations to avoid conflicts. **Turns ON when:** Command is being processed. Command Status Messages MI is 0. **Turns OFF when:** Configuration is first activated. Command Status Messages MI is 1.

	LC Success Bit	**Function in Progress**	**LC Command Status Messages**
Command Issued	0	1	0
Success	1	0	1
Error	0	0	>1

Scan

A Scan FB must be included for every Loadcell Configuration in a Loadcell application. When an application contains more than 1 Loadcell Configuration in the Ladder, the Scan FB displays the Select Name drop-down list, enabling you to link the desired Configuration.

The Scan FB is located on the FB>Loadcell menu.



Note • The Scan FB must be on the left-hand ladder rail.

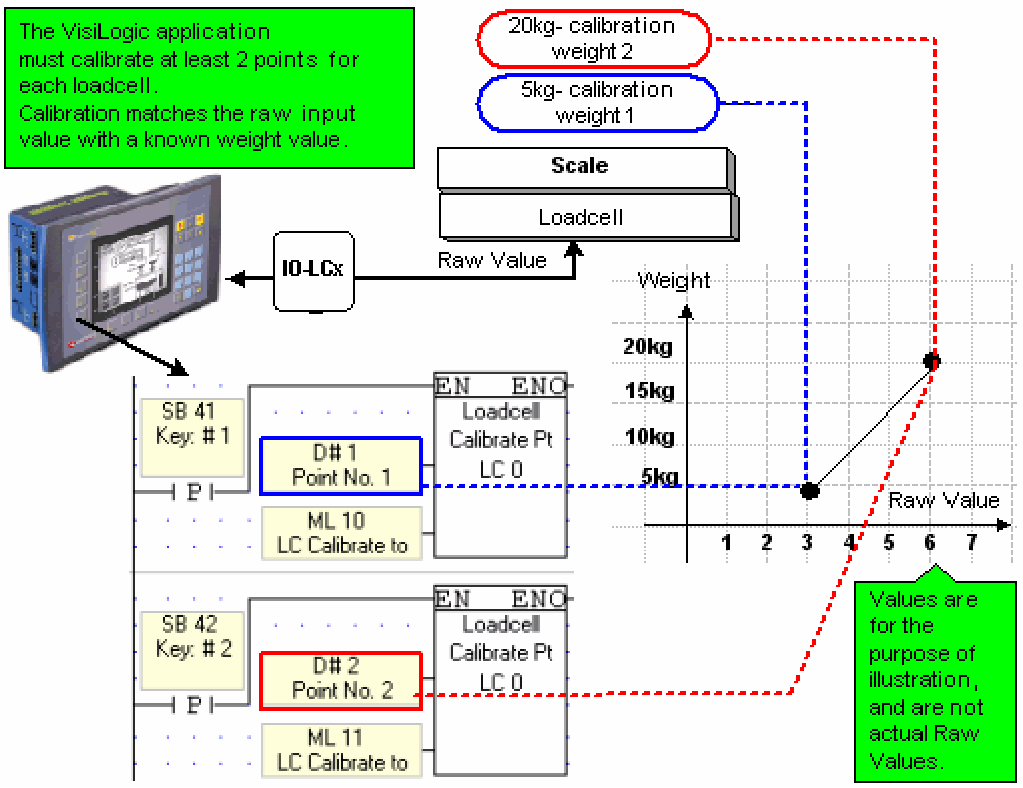
Calibration

Calibration parameters include calibrated points, input range settings, tare and zero values. These may be burned to the module's EEPROM using the Save Calibration FB. Before you can begin to implement a Loadcell application, you must calibrate at least two points, although up to 12 points may be calibrated; all other calibration parameters are optional. However, note that if the application requires you to set Input Range/Gain, you must make these settings **before** you calibrate points. Setting the Input Range/Gain after calibrating points invalidates these points.

Calibrating Points

A Calibration Point matches a Raw Value with a Weight value. These points are used to linearize the input value.

To calibrate points, connect the controller to the loadcell via the I/O-LCx. Initial calibration is generally performed with known weights as shown in the following figure. After calibration has been performed, advanced calibration enables points to be added or edited via the ladder without weight being physically placed on the loadcell.

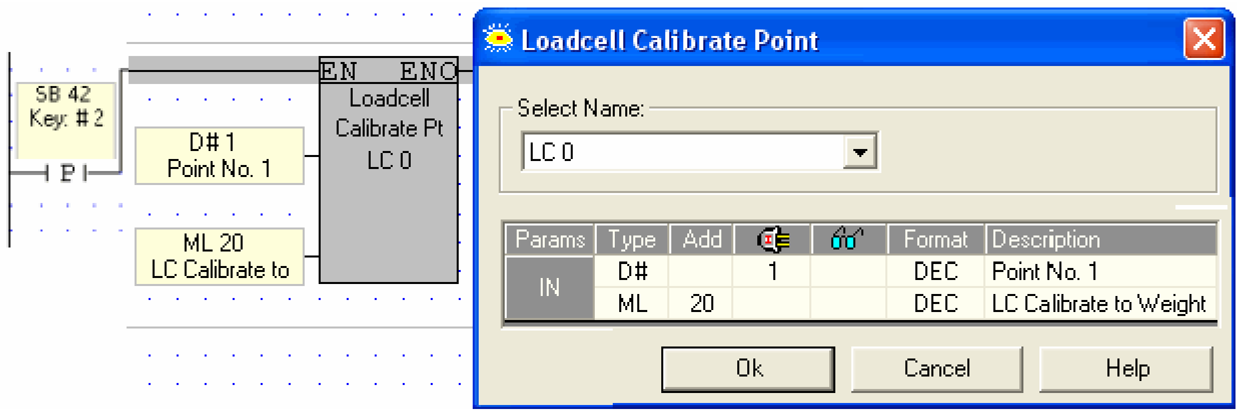

The VisiLogic application must calibrate at least 2 points for each loadcell.
Calibration matches the raw input value with a known weight value.
20kg- calibration weight 2
5kg- calibration weight 1
Scale
Loadcell
Raw Value
IO-LCx
Weight
20kg
15kg
10kg
5kg
Raw Value
1 2 3 4 5 6 7
SB 41 Key: # 1
P
D# 1 Point No. 1
ML 10 LC Calibrate to
EN ENO Loadcell Calibrate Pt LC 0
SB 42 Key: # 2
P
D# 2 Point No. 2
ML 11 LC Calibrate to
EN ENO Loadcell Calibrate Pt LC 0
Values are for the purpose of illustration, and are not actual Raw Values.

Notes •	If the application requires you to set Input Range/Gain, you must make these settings before you calibrate points. Setting the Input Range/Gain after calibrating points invalidates the calibrated points.
•	Zero does not have to be calibrated.
•	Points do not have to be calibrated in any particular order.
•	All calibrated points must be separated by a raw value minimum of 256 (100 Hex).
•	Calibration is an immediate operation; motion is not checked before the operation is carried out.
•	Calibration should be performed with greater accuracy than is required by the application. For example, in an application that requires 100g accuracy, calibrate in units of 10g, then round off the represented value by 10.
•	The highest Calibrated Point weight value should 80–100% of the scale capacity.
•	Calibration cannot be performed if the selected representation

	mode is uV/V.
•	During Calibration, increase filter depth by: - Increasing Settling Time. - Disabling other Loadcells.

Calibrate Point FB

The Calibrate Point FB is located on the FB's>Loadcell>Calibration menu.

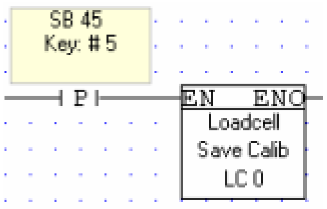


Params	Type	Add			Format	Description
IN	D#		1		DEC	Point No. 1
	ML	20			DEC	LC Calibrate to Weight

Parameter	Type	Function
Point Number	Constant, MI	A Calibration Point matches a weight value with a raw value. The raw value is acquired as an input from the loadcell when the application is run.
Calibrate to Weight	MI, ML	This provides the weight value that is matched to the raw value

Save Calibration

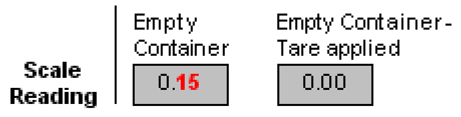
When you save the calibration, calibrated points, tare, zero, and input range are burned into the module's EEPROM memory. This protects the calibration in the event of a power outage, reset, or power-up. To preserve any changes made to calibrated points, input range settings, tare and zero values, use Save Calibration any time these parameters are edited.



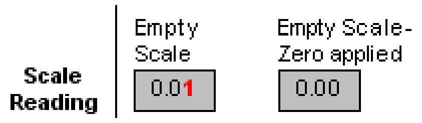
Tare & Zero

Applying Tare and Zero accomplish the same aim: to start a weighing session with a value of zero.

The Tare value may include, for example, the container of the material to be weighed.



If the scale does not read 0 when empty, use Zero to compensate.



When Tare is applied, it is reflected in the net weight.

When Zero is applied, only the gross weight will be zero at the beginning of a weighing session.

Acquire Tare/Zero FB: value read from Loadcell

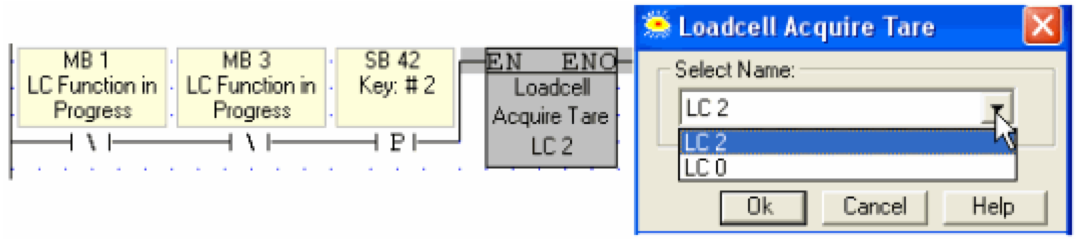
Acquire Tare and Acquire Zero are both located on the FB's>Loadcell>Tare & Zero menu.

Acquire Tare: In this method, the tare value is acquired from the scale. The objects comprising the tare, such as a pallet or materials container, are placed on the scale, and Acquire Tare is activated.

Acquire Zero: The scale must be empty to acquire Zero. Acquire Zero is **not** related to the Auto-Zero Tracking function, which enables the module to compensate for the accumulation of undesired material on the scale in the course of operations.

Note •	Loadcell Name determines from which loadcell the tare/zero will be acquired. After the tare has been acquired, the tare value will be applied to that loadcell.
•	Use the Save Calibration FB to save Tare and Zero values to the module's EEPROM memory.
•	Tare and Zero cannot be acquired when running uV/V mode.
•	If the Motion Band FB is activated, the tare value cannot be acquired until the scale is stable.

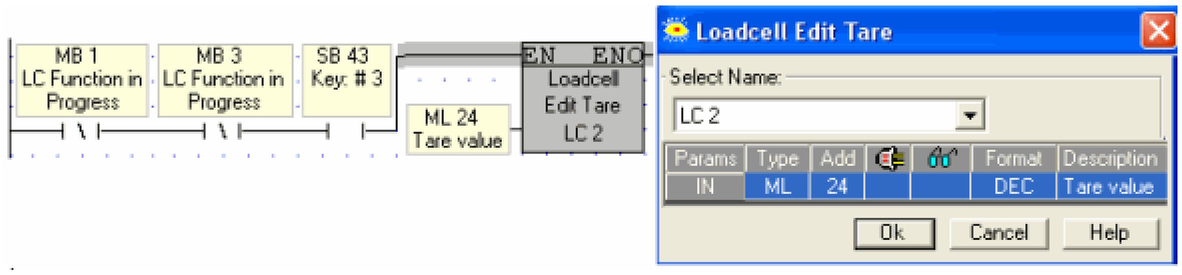
Although only the FBs relating to tare are pictured in the following figures, the figure apply to both Tare and Zero FBs.



<u>**Advanced Tare& Zero Functions**</u>

These FBs are both located on the FB's>Loadcell> Advanced> Tare & Zero menu.

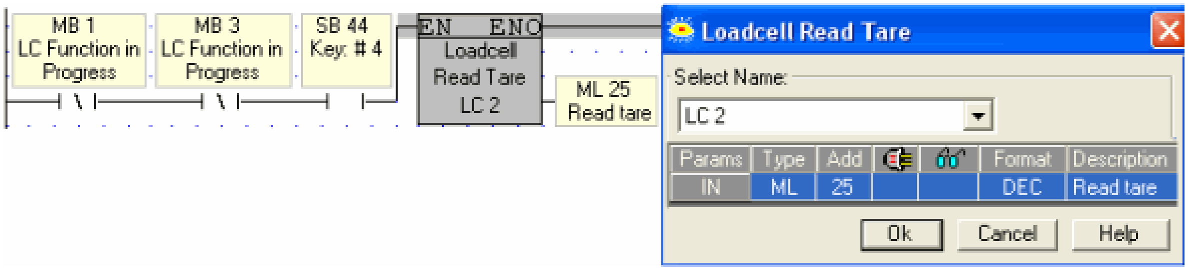
Edit Tare/Zero: value via operand or constant



Params	Type	Add			Format	Description
IN	ML	24			DEC	Tare value

Parameter	Type	Function
Tare/Zero Value	MI, ML, or Constant	Edit Tare/Zero enables the tare value to be acquired from a register or constant value within the controller. You can also use Edit Tare/Zero to change a loadcell's existing tare/zero value.

Read Tare/Zero: reading the current Tare or Zero Value



Params	Type	Add			Format	Description
IN	ML	25			DEC	Read tare

Parameter	Type	Function
Read Tare/Zero	MI, ML	Reads the current tare or zero value applied to the linked loadcell input into the linked register.

Setup

Setup FBs provide additional parameters that you may require for your application. Setup FBs include Motion Band, Filter & Rounding, and Auto-Zero, and are located on the FB's>Loadcell>Setup menu.

Note • The Setup FBs need to be activated only once, at power-up. They are not saved to EEPROM.

Motion Band

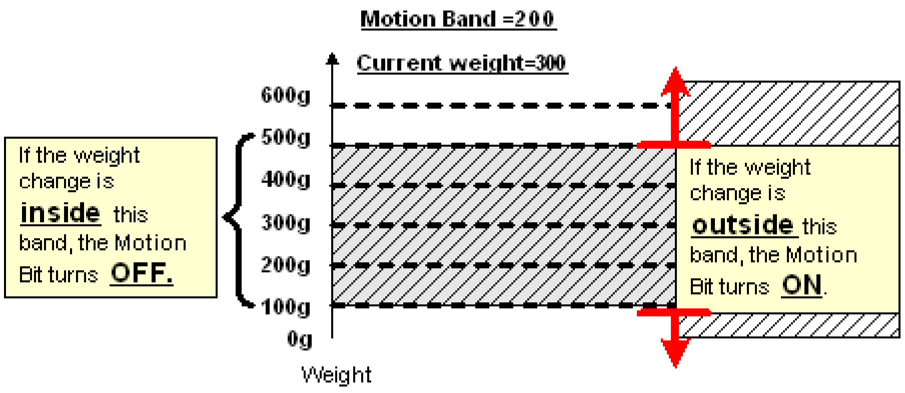
When the weight on the scale changes, the scale needs time to stabilize.

The Motion band determines the amount of weight change the module uses to decide if the scale is in motion.

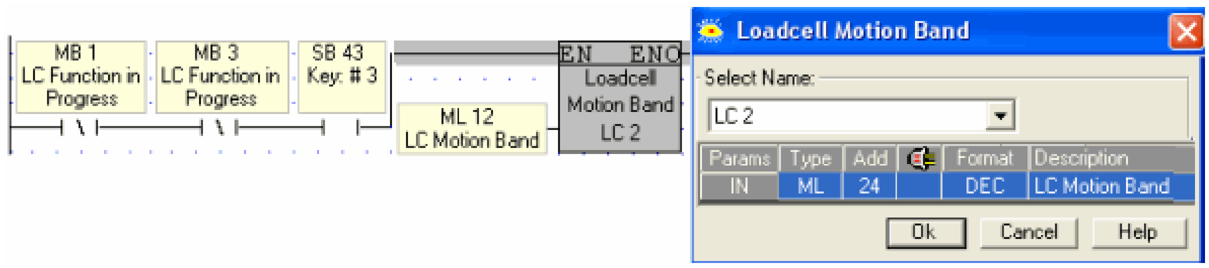
Bit 0, of the MI that is linked to **LC Hardware Status Messages** in Hardware Configuration, is the In-motion indicator. Bit 0 is ON when the scale is in motion, and OFF when the scale is steady.

As the module reads the signals from the loadcell(s) it calculates the weight value. If a weight change falls within the Motion Band, Bit 0 turns OFF.

In the figure below, the in-motion indicator (Bit 0) turns ON when the weight change is below 100 grams, or more than 500 grams. When the weight change falls within the band (100 to 500), Bit 0 turns OFF.


Motion Band =200
Current weight=300
600g
500g
400g
300g
200g
100g
0g
Weight
If the weight change is inside this band, the Motion Bit turns OFF.
If the weight change is outside this band, the Motion Bit turns ON.

Notes •	The FB must be activated in order to apply the Motion Band.
•	The In-motion indication is OFF: - at Power-up - or when the scale is not calibrated.
•	In order for the In-Motion indication to function properly, the filter Active Band must be equal or higher than the In-Motion Tolerance. Refer to the Filter and Rounding function for description and power-up defaults.
•	If the Motion Band is active, the tare/zero values cannot be acquired when the scale is in motion.

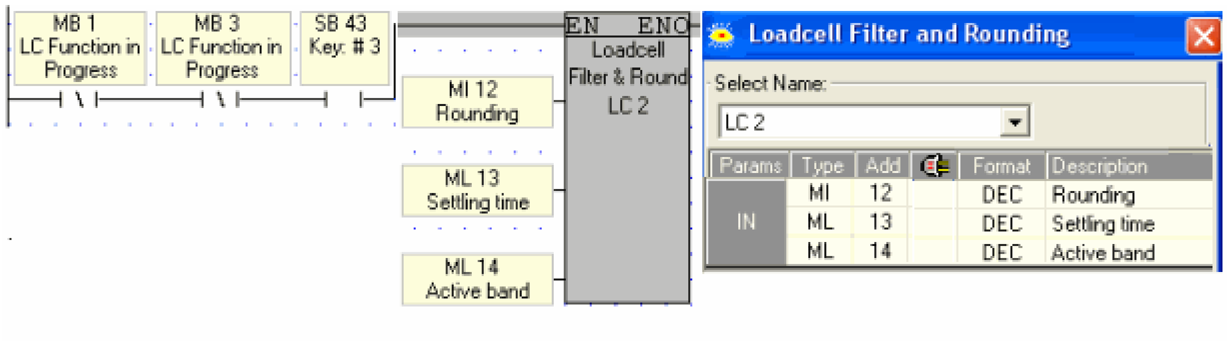


Params	Type	Add		Format	Description
IN	ML	24		DEC	LC Motion Band

Parameter	Type	Function
LC Motion Band	MI, ML, or Constant	Provides a value for the weight band. A weight change that falls outside this band turns the In-motion bit ON. When the weight change is inside the band, the bit turns OFF.

Filter and Rounding

The Filter & Rounding parameters change the default filter parameters and value rounding. Rounding further smooths the loadcell reading.



Params	Type	Add		Format	Description
IN	MI	12		DEC	Rounding
	ML	13		DEC	Settling time
	ML	14		DEC	Active band

Parameter	Type	Function
Rounding	MI or Constant	The number used to round the output values. 0 - Round by 1(Power-up default) 1 - Round by 2 2 - Round by 5 3 - Round by 10 4 - Round by 20 5 - Round by 50 6 - Round by 100 **Note** • Value rounding will not take effect in uV/V and Raw value representation modes.
Settling Time	MI,or Constant	The time, in units of 10msec, that the filter requires to settle to the final reading. **Notes** • The default settling time is 1 second, the minimum time 1 is 12.5 milliseconds, and the maximum is 24 seconds. • A value of zero disables the filter. • Settling time rises with the number of active loadcells. The minimum settling times are: - 12.5ms for one active loadcell. - 675ms for two active loadcells. - 1,012.5ms for three active loadcells. Using a settling time of zero sets the settling time to its minimum value without returning an error.
Active Band	MI, ML, or Constant	The band of weight changes in which the filter is active. The filter is turned off by weight changes that exceed the active band. This allows a rapid response to large weight changes. When the weight changes become smaller than the active band, the filter turns on. An active band of zero forces the filter to be always active. **Notes** • If the Motion Band is on, the filter's Active Band must be equal or higher than the Motion Band.

Auto Zero Tracking

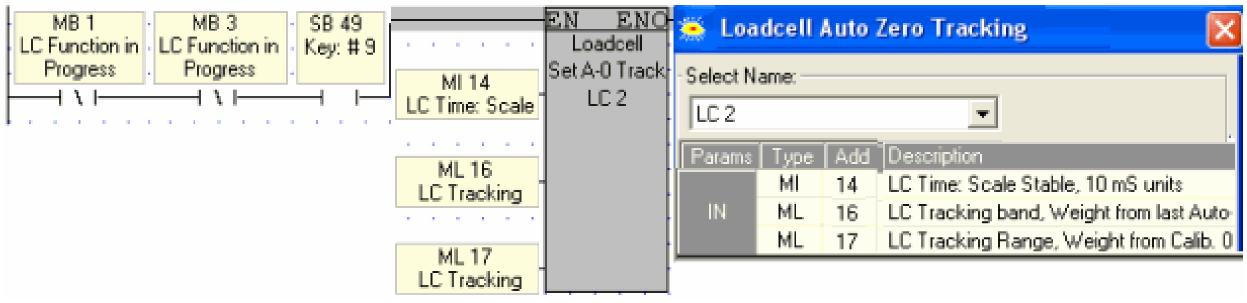
When activated, Auto-Zero Tracking zeros the gross weight according to the conditions you set, enabling the module to automatically compensate for small variations at the zero point, such as those caused by a buildup of litter on the scale platform, or by temperature fluctuations near the scale.

Before Auto-zero Tracking can zero the scale:

- The Motion Band FB must be included in the application.
- The Motion Band must be active.
- The In-Motion bit, Bit 0 of the MI linked to **LC Hardware Status Messages** in Hardware Configuration, must already have turned OFF, indicating that the scale is steady.

Once these conditions have been met, Auto-zero zeros the gross weight.

Notes •	Once Auto-Zero tracking is activated, it stays active until the function is suspended by the application.
•	Auto zero tracking will not function in uV/V representation mode.



LC 2

Params	Type	Add	Description
IN	MI	14	LC Time: Scale Stable, 10 mS units
	ML	16	LC Tracking band, Weight from last Auto
	ML	17	LC Tracking Range, Weight from Calib. 0

Parameter	Type	Function
LC Time: Scale Stable, 10 mS units	MI, ML or Constant	The time in which, in units of 10 mSec, the scale must be stable in order to trigger Auto-Zero Tracking. **Notes** • Initializing this parameter to 0 will turn off Auto Zero Tracking and clear the accumulated offset. • Power-up default: 0 (auto zero tracking is off). • To clear the auto zero tracking offset, initialize this parameter to 0, and then enter a new time value.
LC Tracking band, Weight from last Auto-0	MI, ML or Constant	This determines the maximum distance from the point of the last zero (auto or manual) in which auto-zero tracking is activated [weight units].
LC Tracking Range, Weight from Calib. 0	MI, ML or Constant	This determines the maximum weight from the point of the last calibrated zero in which auto-zero is activated.

Loadcell Operands & Status Messages

Loadcell: Configuration FB

All Loadcell operations run through a Loadcell configuration placed in the control program. Each Configuration is linked to a specific Loadcell input on the I/O expansion module.

Command Status Messages Initialized to 0 when **Loadcell Configuration** is activated.	**MI**	Is reset to 0 when a command is activated. Updated at the end of all operations using the Loadcell Configuration. Indicates error status for all loadcell operations processed by the Configuration Current value always shows the most <u>**recent**</u> error status. Value Message 0 Function in Progress 1 Command carried out successfully 2 I/O Expansion Command Buffer is full, please retry. Can be avoided by using SB 91, I/O Expansion Module--Command Buffer Full, as a condition 3 The I/O expansion module linked to the configuration is busy 5 Timeout Exceeded

		6 Communication Error (I/O module does not exist) 11 Illegal parameter 13 Power supply not connected 16 Scale is currently in motion (is only relevant if In-Motion function is applied) 17 Signal is out of range (this value occurs when the Out of Range bit is ON) 18 Illegal weight (Occurs during calibration, if the raw value of weight being calibrated is too close to the raw value of an already calibrated weight; minimum distance is 256 or 100 Hex) 19 Command not supported in uV/V mode 20 Not calibrated (This value appears when less than 2 points have been calibrated) 21 EEPROM Protection Error (Indicates when too many Save Calibration FBs are run too frequently. Check the activating conditions for the Save Calibration FB, and whether your application contains loops)	
LC Success Bit	**MB**	**Turns ON when:** Configuration is first activated. Command is successfully processed. Command Status Messages MI is 1.	**Turns OFF when:** Command Status Messages MI is 0. Function in Progress MB is ON, indicating that a command is being processed.
Function in Progress	**MB**	**Turns ON when:** Command is being processed. Command Status Messages MI is 0.	**Turns OFF when:** Configuration is first activated. Command Status Messages MI is 1.

Loadcell Hardware Status Messages (Hardware Configuration)

The MI linked to the parameter **LC Hardware Status Messages** in Hardware Configuration provides a bitmap for the following messages.

Bit#	**Description**	**Turns ON when:**	**Turns OFF when:**
0	Scale motion Only relevant if Motion Band FB is included in application and activated	Scale is in motion	At Power-up When Scale is steady
1	Input Value Range Linked to I/O module's Out of Range LED indicator	Input value is out of range Possible causes: 1 or more signal wires are disconnected A/D input voltage is out of range	Input value is in range
2	Input Value Validity	Input Value is invalid Possible causes: Channel is temporarily disabled, via the Disable all other Loadcells FB Bit is ON at Power-up until the first input value is received from the loadcell	Input Value is valid
3	Loadcell Calibration Status	When less than 2 points are calibrated	At least 2 points are calibrated
4	Input Power Supply Status Linked to I/O module's Out of Range LED indicators	No Power	Power Supply OK
		When the input power is not supplied, the indicators blinks rapidly	

Note • Bits 6 & 7 are linked to Outputs 0 & 1, located on the I/O module. Bit 6 is related to Output 0, Bit 7 to Output 1.
Bits 6 & 7 can be used to monitor the setpoint output's status from within the Ladder application.
The I/O module itself controls the setpoint function of the outputs. The module turns the outputs ON and OFF when the current loadcell input value reaches setpoint. Since the function is based in the firmware of the expansion module, when the output's status changes as a result of reaching/departing from setpoint, the status change is not registered by the Ladder application.
Examples

When setpoint output 1 is assigned to load cell channel 0, Bit 7 of load cell 0 status will indicate the state of output 1.
When setpoint output 0 is assigned to load cell channel 2, Bit 6 of load cell 2 status will indicate the state of output 0.

6	Setpoint Status, Output 0	Output 0 is ON	Output 0 is OFF
7	Setpoint Status, Output 1	Output 1 is ON	Output 1 OFF

SB91	I/O Exp. Module--Command buffer is full	ON when commands **cannot** be sent to the I/O module.	OFF when commands **can** be sent to the I/O module..

Advanced Loadcell Functions

Advanced-Calibration

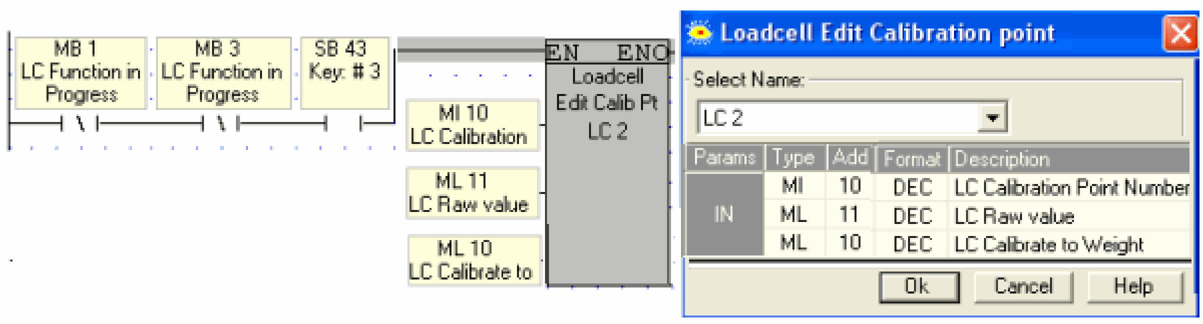
These FBs are all located on the FB's>Loadcell> Advanced> Calibration menu.

Note •	The Edit and Read Calibration Point functions contain parameters which accept the raw and weight values from existing Calibration Points. These parameters must be linked to the same type of registers, MI or ML, used to calibrate the original Calibration Point.

Edit Calibration Point: value via operand or constant

A Calibration Point matches a raw value to a weight value. You can use Edit Calibration Point to change these values.

Notes •	The loadcell will stop functioning if deleting a point causes the number of Calibration Points to be less than 2.
•	To prevent your changes from being lost at power-up, reset, or in the event of power outage, use the Save Calibration FB to burn the changes to the module's EEPROM.



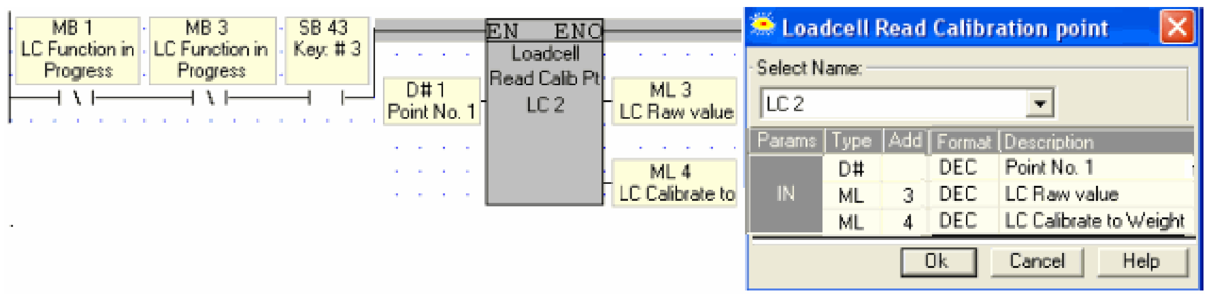
Params	Type	Add	Format	Description
IN	MI	10	DEC	LC Calibration Point Number
	ML	11	DEC	LC Raw value
	ML	10	DEC	LC Calibrate to Weight

Parameter	Type	Function
LC Calibration Point Number	MI or Constant	The number of the Calibration Point that you wish to edit.
LC Raw Value	MI, or ML	This register contains the new raw value for the Calibration Point.
LC Calibrate to Weight	MI, or ML	This register contains the new weight value for the Calibration Point.

Read Calibration Point

Use this function to read the current raw and weight values of a Calibration Point.

Note • If the point being read is not in use, both returned values will be -32768 (0x8000) for integer and -8388608 (0x800000) for long.



Select Name: LC 2

Params	Type	Add	Format	Description
IN	D#		DEC	Point No. 1
	ML	3	DEC	LC Raw value
	ML	4	DEC	LC Calibrate to Weight

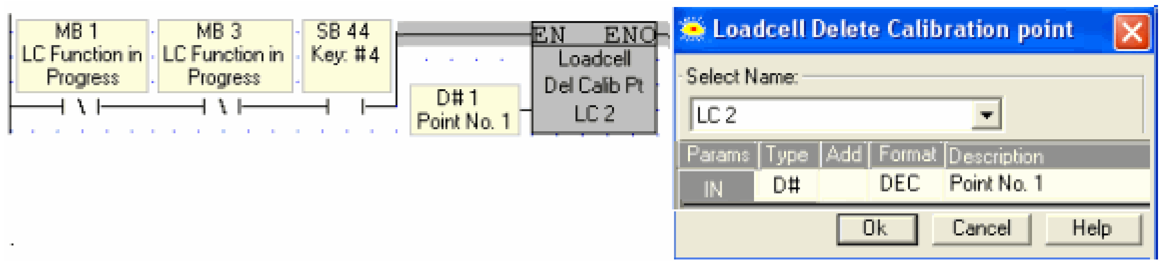
Parameter	Type	Function
LC Calibration Point Number	MI or Constant	The number of the Calibration Point that you wish to read.
LC Raw Value	MI, or ML	This register contains the current raw value of the Calibration Pointt.
LC Calibrate to Weight	MI, or ML	This register contains the current weight value of the Calibration Point.

Delete Calibration Point

This deletes a Calibration Point.

Notes • The loadcell will stop functioning if a deleted point causes the number of Calibration Points to be less than 2.

• To prevent your changes from being lost at power-up, reset, or in the event of power outage, use the Save Calibration FB to burn the changes to the module's EEPROM.



Select Name: LC 2

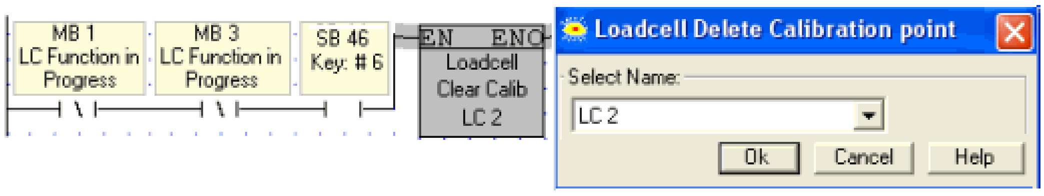
Params	Type	Add	Format	Description
IN	D#		DEC	Point No. 1

Parameter	Type	Function
LC Calibration Point Number	MI or Constant	The number of the calibrated point that you wish to delete.

Clear Calibration

This deletes all calibrated points, tare, zero, and input ranges from the module's memory. However, Clear Calibration does not erase the values from the EEPROM. They may be retrieved from the EEPROM by resetting the controller.

To delete all values from the EEPROM, run Clear Calibration followed by Save Calibration.

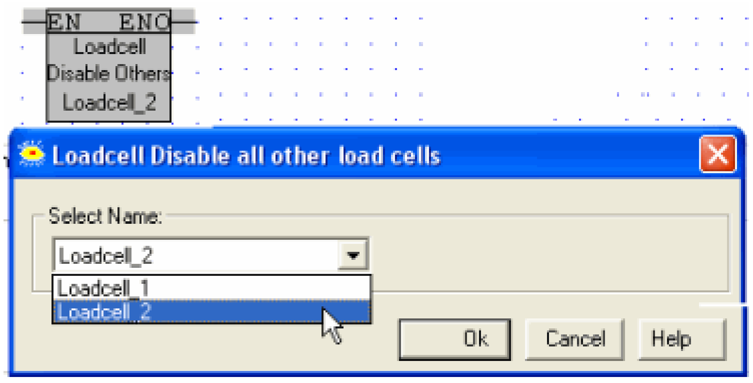


Enable/Disable all other load cells

Disable All Other Loadcells disables all loadcells in the expansion module except for the loadcell selected in the FB.

During Calibration, the Disable All Other Loadcells FB can be used to increase filter depth for a specified settling time by eliminating the delay caused by channel change (approx. 300ms) and thus ensuring faster and more accurate calibration.

To prevent channel changing from wasting settling time, use this function to disable all other loadcells except for the one you are currently calibrating.



Enable All Other Loadcells re-enables all loadcells in the expansion module.

Notes • After this FB is run, and the Command Status Messages MI linked to the selected Loadcell Configuration indicates 1, updated values for all of the enabled channels are already available at their linked operands. This indication can be used to trigger a process, such as calibration.

•	Disabled Loadcells: the Hardware Status Messages MI linked to the selected Loadcell Configuration The status bit "Value not valid" will rise in the disabled load cells' status word.

Setpoint

Each digital output located on the I/O module is associated with a setpoint. The I/O module itself controls the setpoint function of the outputs. The module turns the outputs ON and OFF when the current loadcell input value reaches setpoint. Setpoint activity is therefore not linked to the program scan. Each output may be assigned a setpoint.

Since the function is based in the firmware of the expansion module, when the output's status changes as a result of reaching/departing from setpoint, the status change is not registered by the Ladder application. To monitor the outputs' status, this MI provides a bitmap indicating status messages; Bit 6 is related to Output 0, Bit 7 to Output 1.

Therefore, use Bits 6 & 7 of the **LC Hardware Status Messages** MI to monitor the outputs' status, from within the Ladder application.

Note •	Once the Setpoint is activated, it cannot be changed by setting the output via the Ladder application.
	The setpoint remains OFF, regardless of its N.O./N.C.setting, when the loadcell input value is: - invalid (i.e., powered off, LC disabled, out of range, loadcell not calibrated. - In uV/V mode.

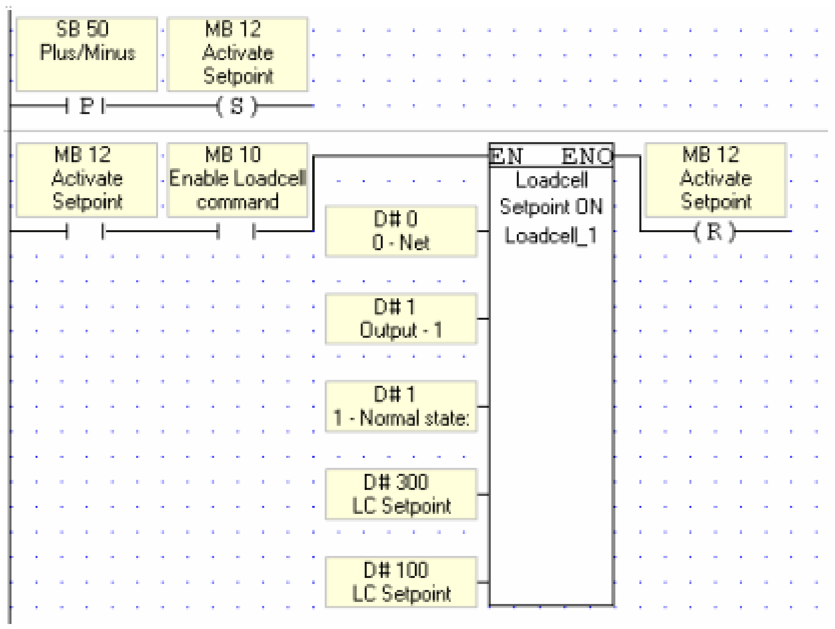
Examples

- When setpoint output 1 is assigned to load cell channel 0, Bit 7 of load cell 0 status will indicate the state of output 1.
- When setpoint output 0 is assigned to load cell channel 2, Bit 6 of load cell 2 status will indicate the state of output 0.

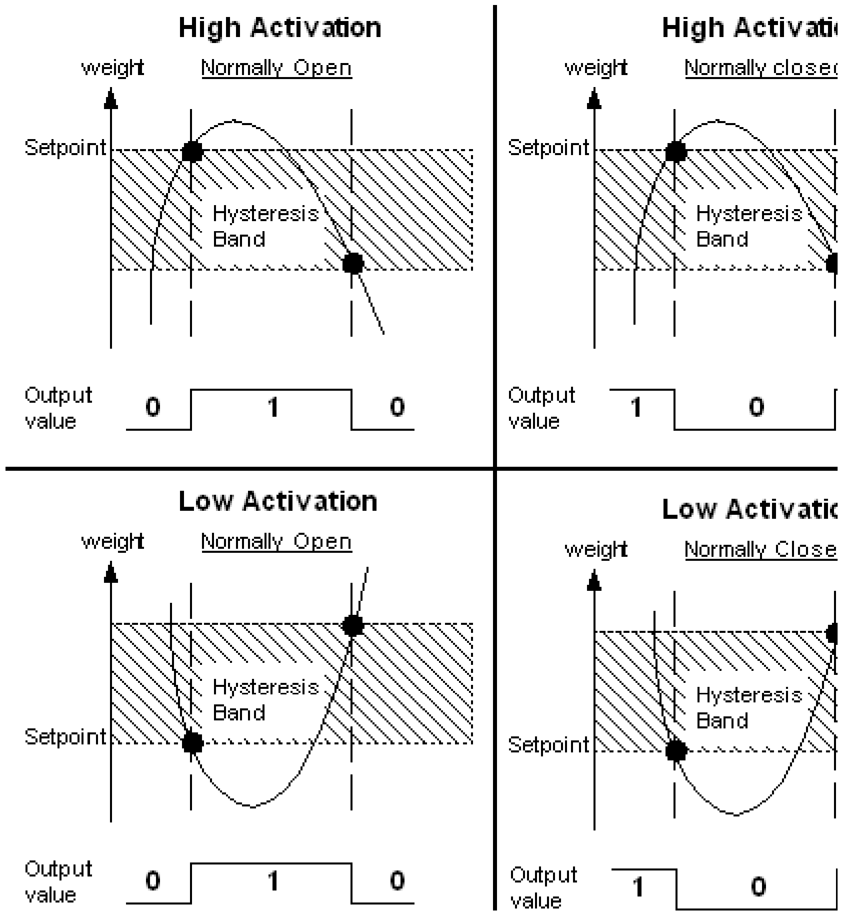
Bit	Description	Turns ON when:	Turns OFF when:
6	Setpoint Status, Output 0	Output 0 is ON	Output 0 is OFF
7	Setpoint Status, Output 1	Output 1 is ON	Output 1 OFF

Set and Activate Setpoint

Use this FB to implement a desired setpoint.

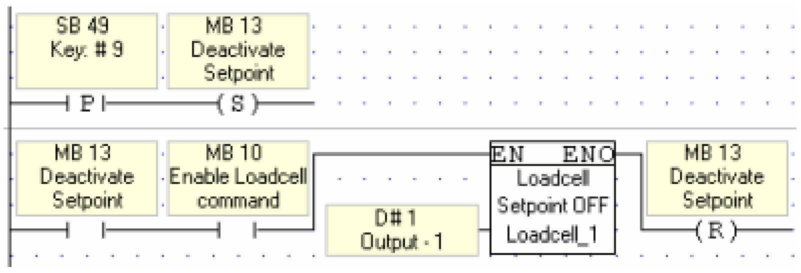


Parameter	**Type**	**Function**
LC Value Mode	Constant	Set the input value mode for the setpoint: 0 - Net 1 - Gross 2 - Net Min 3 - Net Max
LC Output Number	Constant	Select output: Output - 0 Output - 1
LC Setpoint Type	MI, ML, or Constant	Select setpoint type: 0 - Normal state: Open Activation: Low 1 - Normal state: Open Activation: High 2 - Normal state: Closed Activation: Low 3 - Normal state: Closed Activation: High

		 High Activation weight Normally Open Setpoint Hysteresis Band Output value 0 1 0 High Activati weight Normally close Setpoint Hysteresis Band Output value 1 0 Low Activation weight Normally Open Hysteresis Band Setpoint Output value 0 1 0 Low Activatic weight Normally Close Hysteresis Band Setpoint Output value 1 0
LC Setpoint Value	MI, ML, or Constant	The actual value assigned to the setpoint.
LC Setpoint Hysteresis	MI, ML, or Constant	Sets a band in which the output will not chatter due to overshoot or vibrations.

Deactivate Setpoint

Use this to suspend the activity of a particular setpoint.

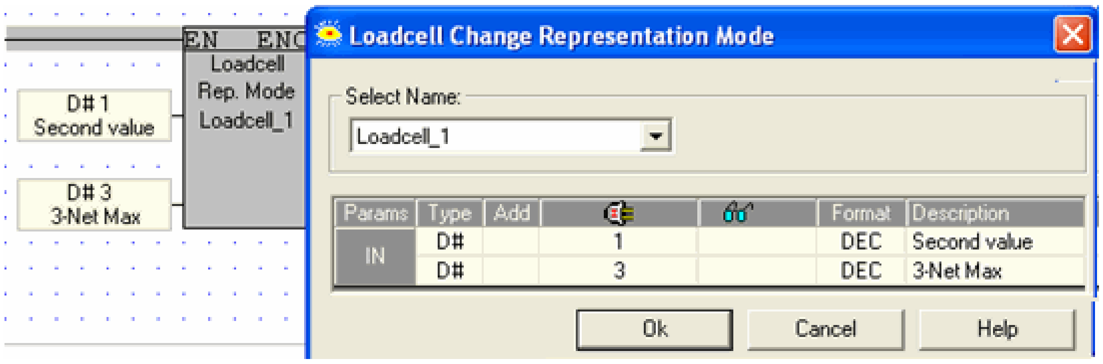


Change Representation Mode

During hardware configuration, under Number of values, you select whether to use one or two values. When you select a register for the Address: Value(s) parameter, selecting two values means that the register immediately following the register you select is used for the second value. The default representation mode for the first register is Net Weight, and Gross Weight for the second register.

Via Change Representation mode, you can 'read' the value as:

- Gross weight
- Net weight
- Net Min. Weight
- Net Max. Weight
- Scaled to mV
- Raw Value



Params	Type	Add			Format	Description
IN	D#		1		DEC	Second value
	D#		3		DEC	3-Net Max

Parameter	Type	Function
LC Value to Represent	Constant	Select the value for which you want to change the representation mode.
LC Representation Mode	MI or Constant	Select the desired representation mode. 0 - Net (Gross if no Tare) (Power-up default for 1st value) 1 - Gross (Power-up default for 2nd value) 2 - Net Min 3 - Net Max 6 - uV/V 7 - Raw value

Note • When, after Change Representation Mode runs, the LC Command Status Messages MI returns '1', the requested value is already in its linked operand. This means that you can use the '1' status to trigger a process which relies on this specific representation value.

• The mV/V representation mode uses the default calibration. Therefore:

- The mV/V rep. mode indicates the actual applied differential input voltage in micro-volts per every volt of the excitation, regardless of the user-selected input range and DAC (offset) compensation.
- Setting **one** of the values representation modes to mV/V will force **both** values to be represented in mV/V (the rep. mode of the other value will not be overwritten).

	● It takes approximately 330msec to change between mV/V and other different representation modes.
•	The A/D raw value is affected by the user-selected input range (gain and DAC (offset) compensation). To cancel this effect, use the Clear Calibration command to set default calibration. To return to the last saved calibration, reset the controller (no need to re-power-up neither the unit nor the controller).

Reset Net Min/Max Values

Resets the Net Minimum value to positive full-scale, and the Net Maximum value to negative full-scale.

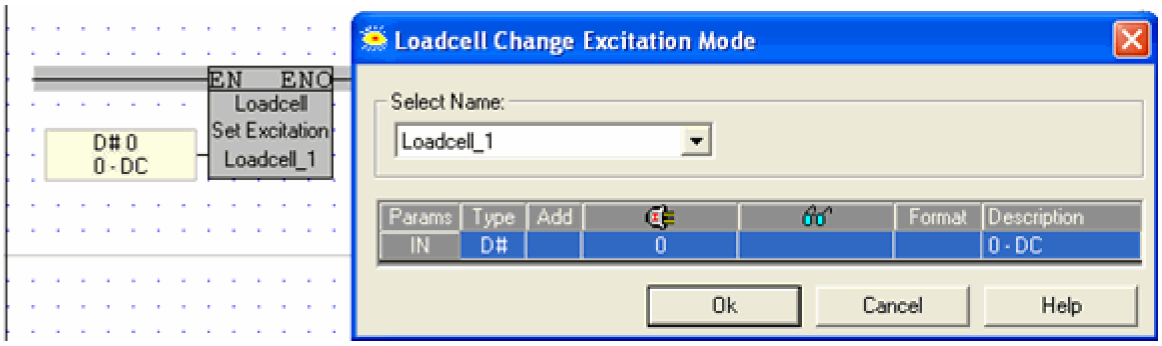
As soon as the scale becomes stable, meaning that the In-motion bit is OFF, the Net Min and Max values will be set to the net value.

A Net Min and Max reset occurs also at power-up.

Change Excitation Mode

Use this FB to temporarily change the excitation supplied to the loadcell.

This method is intended to use only for diagnostic purposes,such as when using a DC milli-voltmeter.



Select Name: Loadcell_1

Params	Type	Add			Format	Description
IN	D#		0			0 - DC

Note •	Changing the excitation mode may add an offset to the A/D measurements. Therefore, the system should be calibrated using the same excitation mode the loadcell will work with.
•	In general, the working excitation mode should be set via Hardware Configuration.
•	The Change Excitation command overrides the hardware configuration excitation setting until the next system reset / power-up.
•	Changing excitation mode may cause a momentary conversions-break (about 300msec) due to filter reset.

Parameter	Type	Function
LC Excitation mode	MI or Constant	0=DC 1=AC

Input Range

The Input Gain parameter sets the amplification range for the input signal.

The Offset parameter is generally used to compensate for the deadload; particularly in cases where the combined weight of deadload and payload exceed the A to D converter input range.

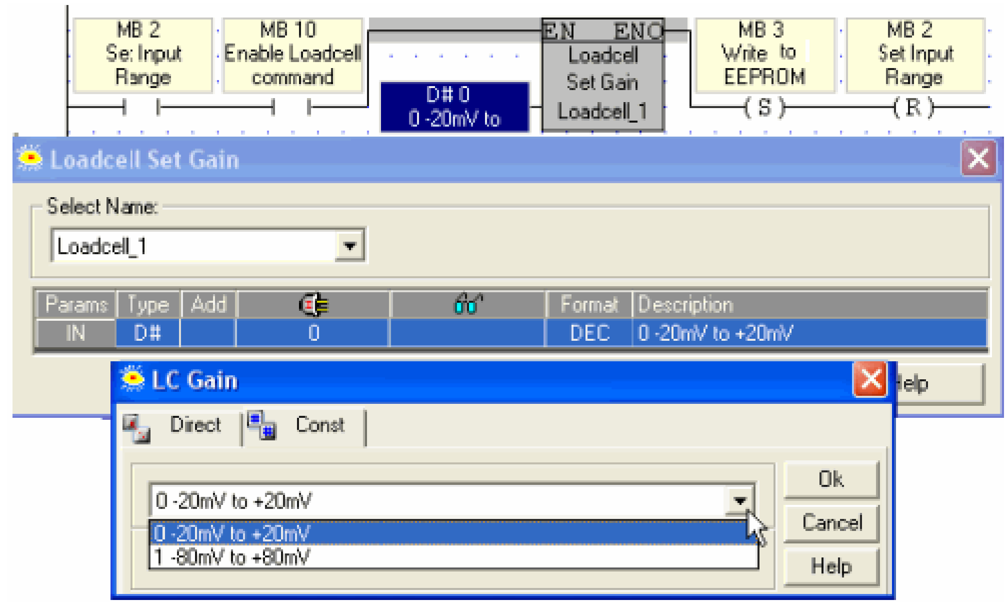
Input Range and Offset are considered part of the loadcell's calibration. To burn these values into the module's EEPROM memory and protect them in the event of a power outage, use the Save Calibration FB.

Notes •	Changing Gain or Offset requires you to recalibrate and save all calibrated points.
•	If the application requires you to set Input Range/Gain, you must make these settings **before** you calibrate points. Setting the Input Range/Gain after calibrating points invalidates the calibrated points.
•	Offset values out of the ±31 range will be truncated and no error will be returned.
•	The uV/V rep. mode uses its own input range settings and therefore is not affected by the command.

Set Gain FB

This FB limits the input range. The gain is applied to the signal **after** offset compensation.

Setting the Set Gain MB to 0 limits the input range to ±20mV (Default setting), setting it to 1 (or any other value) limits the input range to ±80mV.



Loadcell Set Gain

Select Name: Loadcell_1

Params	Type	Add			Format	Description
IN	D#		0		DEC	0 -20mV to +20mV

LC Gain

0 -20mV to +20mV
0 -20mV to +20mV
1 -80mV to +80mV

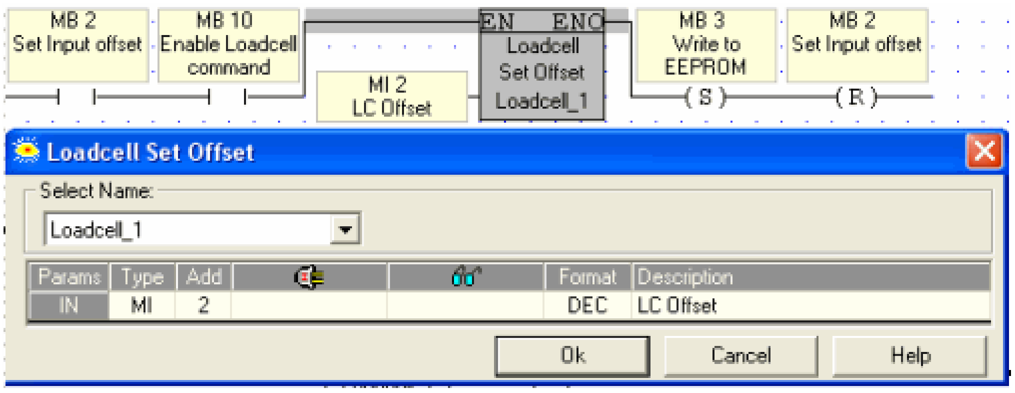
Set Offset

This FB sets the offset compensation, which is applied to the input signal **before** the gain. By default, the offset is set to 0mV (no offset).

Possible values are in the range of ±31, where: 1LSB ≈ 0.5mV/V (= 2.5mV at exactly 5V excitation). Hence, the maximum offset compensation is ±15.5mV/V (= ±77.5mV at exactly 5V excitation).

To calculate the offset value, measure the differential voltage at the input, between the -SG and +SG terminals, and then calculate the offset value according to 1LSB ≈ 0.5mV/V.

If, for example, the differential voltage at the input is 10mV, use -4 as the offset value.



Loadcell Set Offset

Select Name: Loadcell_1

Params	Type	Add			Format	Description
IN	MI	2			DEC	LC Offset

Read Gain

Reads the input range Gain.

Read Offset

Reads the input range Offset.

Filter

Filter Overview

Filter, located on the FB menu, enables you to take from 4 to 16 values and calculate an average. You can influence the average by configuring the function to:

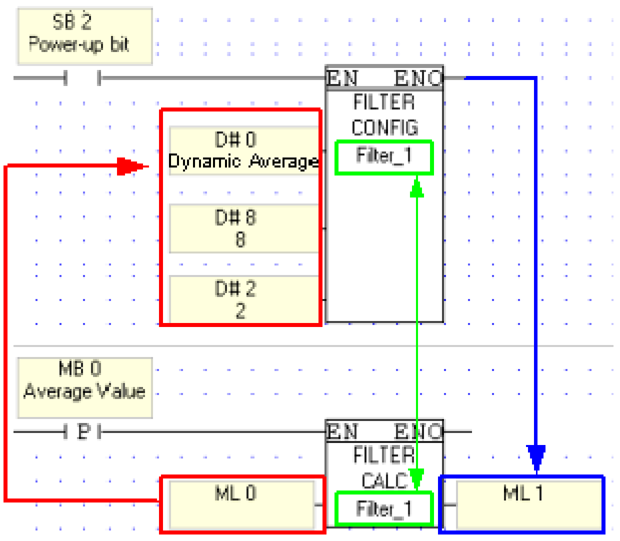
- Discard a user-defined number of maximum & minimum input values
- Weight the average according to the order of the input values (FIFO).

Note •	The values are not allocated memory, but are stored and calculated independently within the function block.
•	Certain analog inputs can also be filtered via Hardware Configuration settings.

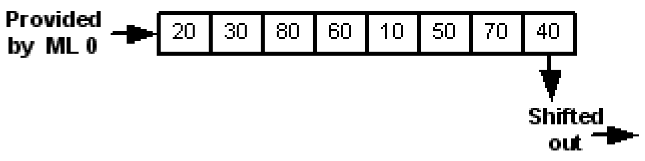
How Filter works

The Filter Configuration holds the parameters that determine how the input values are averaged. The input value is taken from the linked Filter Calculate function.

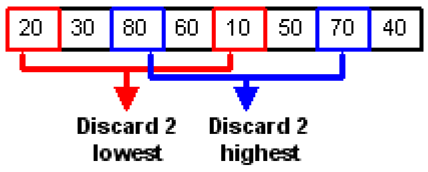
In the figure below, Filter Calculate is linked to the configuration Filter_1. In this configuration, Parameter 1: Filter type is Dynamic Average; Parameter 2: Number of Values is 8; and Parameter 3: Discard Values is 2.



ML 0 provides the input values. Each time Filter: Calculate is activated, the value in ML 0 is copied to the vector of the linked Filter Configuration,. In this example, 8 values are collected. As each new value is input, the oldest value is shifted out.



Since Discard Values is 2, the 2 lowest and 2 highest values are discarded. The remaining values are summed, and then averaged (30+40+50+60=180/4). The result is 45, which is output to ML 1.



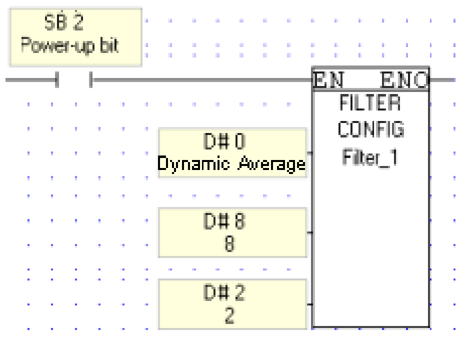
FB Operations

Filter: Configuration

Filter: Calculate

Filter: Configuration

The Filter Configuration holds the parameters that determine how the input values are averaged, therefore it must be activated before the Filter: Calculate function is called.



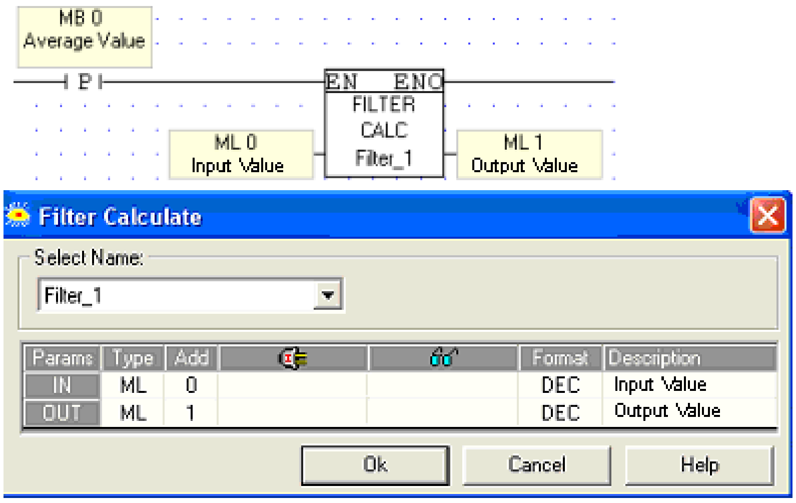
Parameters	Type	Function
Filter Type: Calculates the average of all the current input values	Dynamic Average	Each input value is given equal weight.
	Order-weighted Average	More weight is given to the newer input values. (FIFO)
Number of Values	4-16	Defines the values in the vector as well as the divisor for calculating the average.
Discard Values	0-4	Removes the minimum and maximum input values according to the selection. For example, selecting the number '3' cause the 3 lowest and 3 highest values to be discarded before the average is calculated.

Filter: Calculate

Each time the Calculate function's activating condition is turned ON, the function's Input parameter is copied to the vector of the linked Filter

Configuration, the average is calculated, and the result is placed in the linked output operand.

Note •	The output value is overwritten each time the function is activated.
•	The vector is not allocated PLC memory. The values are stored in the function



Params	Type	Add			Format	Description
IN	ML	0			DEC	Input Value
OUT	ML	1			DEC	Output Value

Accelerate

Accelerate Overview

Accelerate causes a register value to increment or decrement within a set range, at an accelerating or decelerating rate. This rate increases/decreases exponentially as long as the function receives power flow.

The value to be incremented/decremented is set in the Configuration, together with the other parameters that set the rate of change.

Accelerate operations are located on the FBs toolbar.

FB Operations

Configuration

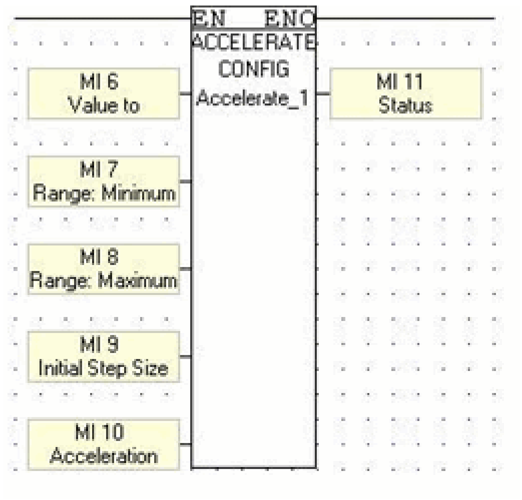
Increment/Decrement

Configuration

The Configuration should be called as a Power-up task. The function sets the parameters which determine the value to increment or decrement, the range for that value, the initial step size, and the factor of acceleration/deceleration.

The rate of acceleration is calculated as follows:

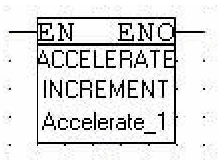
Rate of Acceleration = Acceleration factor * Initial Step Size * Time (where time is the duration of the time period that Accelerate is active)



Parameter	Type	Function
Value to Increment/Decrement	MI,ML, DW, or	This is the register value that is incremented ('accelerated') or decremented ('decelerated).
Range: Minimum Value	MI,ML, DW, or Constant	The value being incremented or decremented cannot fall below this value. **Note** • Both minimum and maximum values must be assigned to the same operand type.
Range: Maximum Value	MI,ML, DW, or Constant	The value being incremented or decremented cannot exceed this value.
Initial Step Size	MI or Constant	When the function is activated, this determines the step size during the first 3 seconds of Increment/Decrement. One step is made each seconds. After three seconds has elapsed, the function uses the Initial Step Size as a factor in calculating the rate of acceleration. **Note** • This value cannot exceed 255.
Acceleration Factor	MI or Constant	The factor used in the acceleration/deceleration formula. **Note** • This value cannot exceed 255.
Status	MI	Bit Message 0 No error 1 Range: Minimum Value and Range: Maximum Value have not been assigned to the same operand type. 2 Range: Minimum Value is higher than or equal to Range: Maximum Value. 3 Accelerated value exceeds Range: Maximum Value. 4 Accelerated value is lower than Range: Maximum Value. 5 Initial Step Size exceeds 0 - 255 (byte size). 6 Acceleration Factor exceeds 0 - 255 (byte size).

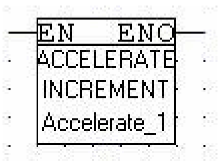
Increment/Decrement

Increment



When this is active, the Value to Increment/Decrement increases according to the rate of acceleration. The longer Increment is active, the faster the rate of increase.

Decrement

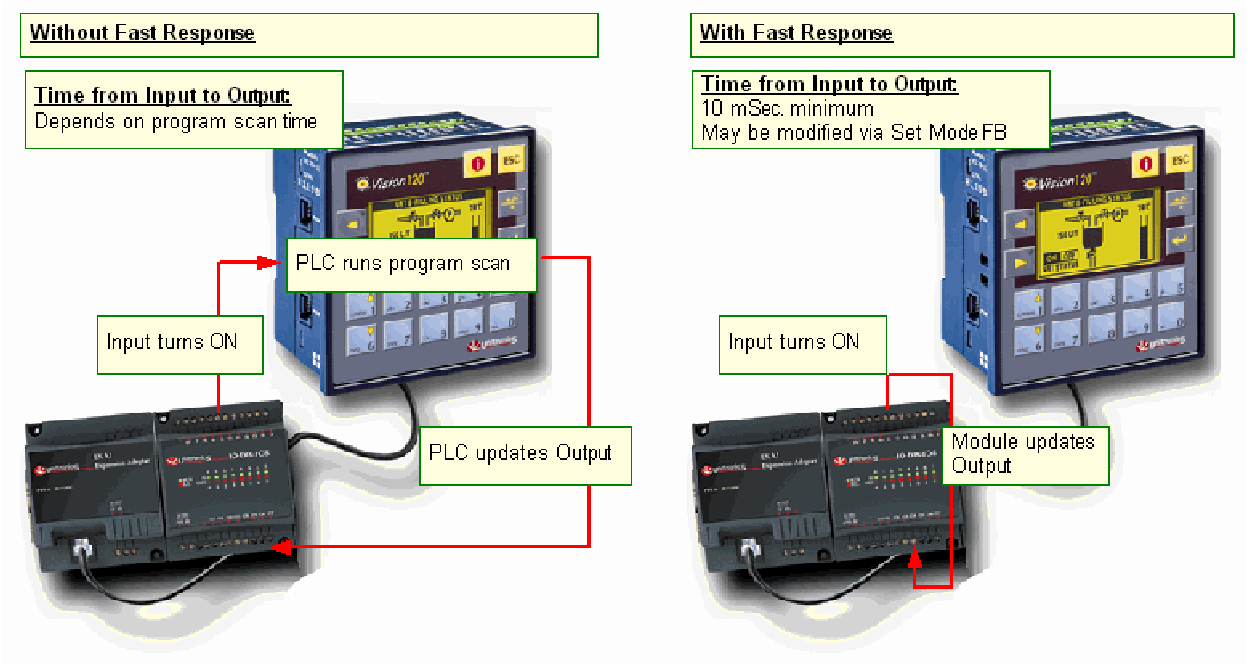


When this is active, the Value to Increment/Decrement decreases according to the rate of acceleration. The longer decrement is active, the faster the rate of decrease.

Fast Response

Fast Response Overview

Fast Response functions are intended for applications that require very fast reaction times. The functions may be used with the I/O Expansion Module I/O-DI8-TO8. In conjunction with Fast Response FBs, the module can use input state to control the state of an output. The control is direct, internal to the module, and is independent of the program scan.

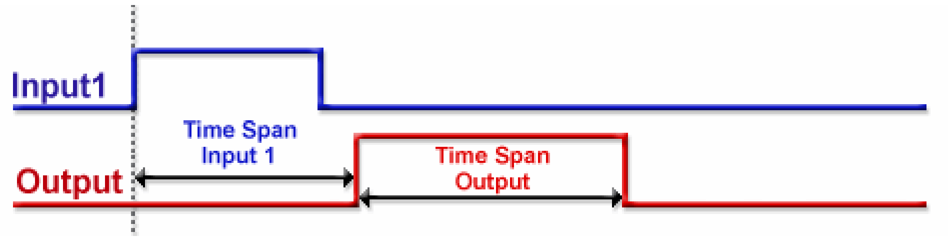


The module can be set to work in four different modes via the Set Mode operation. In addition, the Set Counter operation may be used to store a value or reset the high-speed counter on the module.

Fast Response: Mode Overview

Mode 1: Single Input

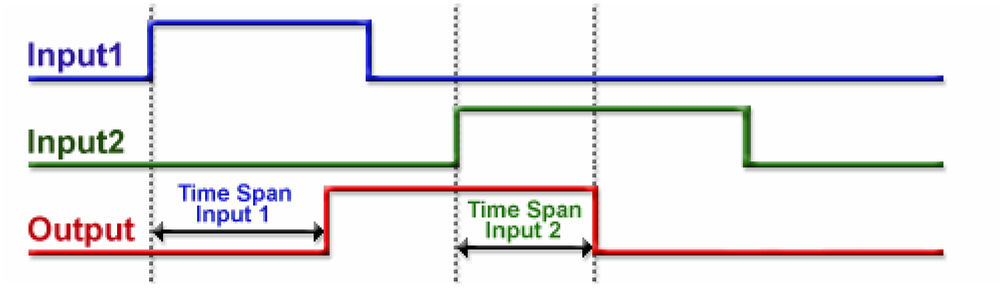
- Register when a selected input, **Input 1,** turns ON or OFF.
- Wait the time span defined for **Input 1**(units of 1mSec), then change the state of the corresponding output.
- Maintain the state of the output for a defined time span, and then return the output to its original state.



Mode 2: Two Inputs

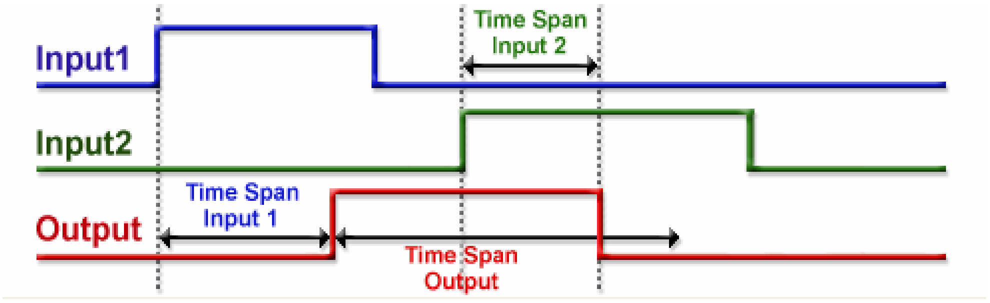
- Register when **Input 1** turns ON or OFF.

- Wait the time span defined for **Input 1**, then change the state of the corresponding output.
- Register when **Input 2** turns ON or OFF.
- Wait the time span defined for **Input 2**, then return the output to its original state.

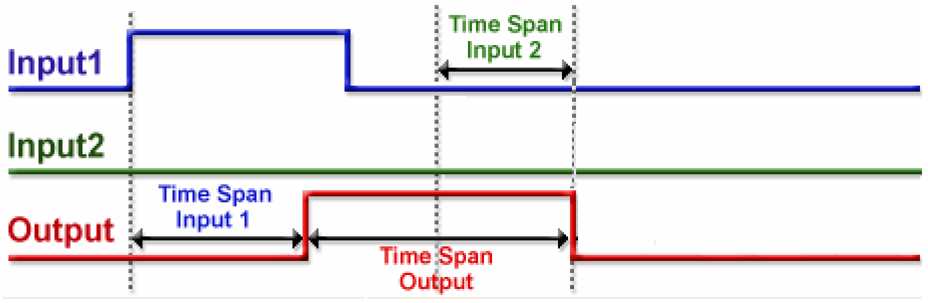


Mode 3: Two Inputs + Output Timer

- Register when **Input 1** turns ON or OFF.
- Wait the time span defined for **Input 1**, then change the state of the corresponding output.
- Register when **Input 2** turns ON or OFF.
- Wait the time span defined for **Input 2**, then return the output to its original state.



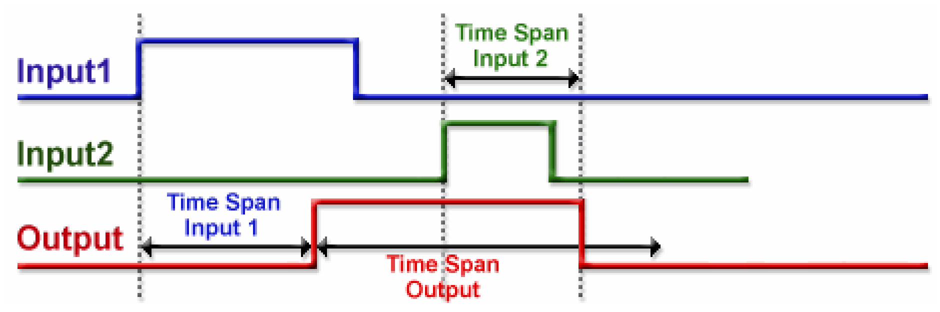
- If, however, **Input 2** does **not** change state within the time span defined for the **corresponding output**, the output state changes.



Mode 4: Two Inputs + Output Timer +T2

This mode is identical to Mode 3, with an added feature:

- The time span defined for **Input 2** has an **additional** function. It determines the time span from the change of the output state, during which **Input 2** is ignored.



Using Fast Response: Overview

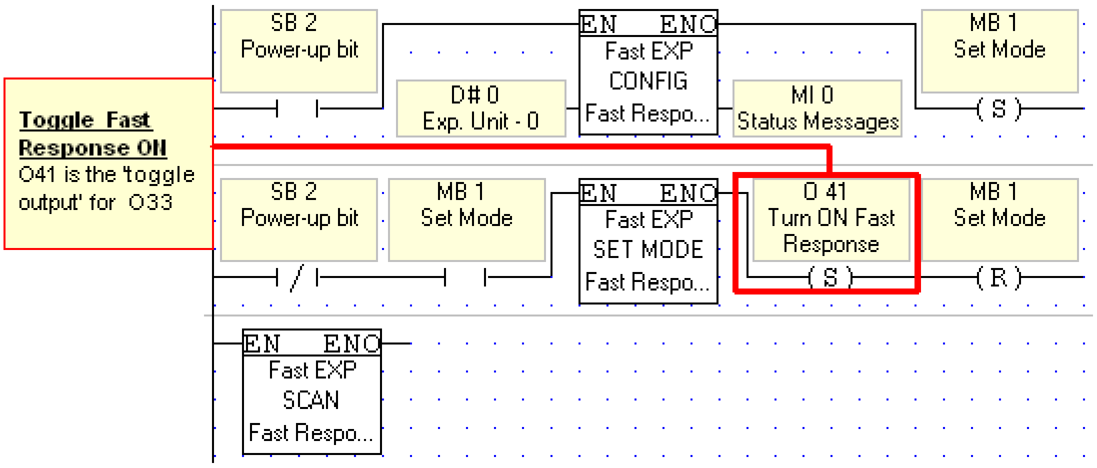
A Fast Response application must contain the following elements:

- Configuration
  Use this to select the I/O Expansion Module for Fast Response
- Set Mode, to select the Fast Response mode and the input or inputs. Note that selecting an input automatically selects the corresponding output; for example, selecting Input 35 automatically selects Output 35.
- Scan

'Fast Response' Toggle

To enable an output to directly respond to an input and bypass the program scan, you must toggle the output into 'Fast Response' mode by turning the appropriate 'trigger output' ON. To calculate the address of this output, check the section below: **Determining the 'toggle output' address.**
To suspend Fast Response and return control to the PLC program, turn the mirror output OFF.

In the application shown below, I 33 is selected as the input. This means that O33 is automatically selected, and that O41 is the 'toggle output' that you use to toggle in and out of Fast Response mode.



Determining the 'toggle output' address

Hardware Configuration shows a DIN rail that holds the I/O Expansion models for your application. The position of the module determines the address of an output. Each module has a group of 16 assigned outputs, no matter how many outputs it actually has. The address of an I/O indicates its actual physical location. This address relates to both the position of the expansion module in the system, and to the position of the I/O on that module.

The formula that is used to calculate the address of an I/O is:

(32 + x) + (16 + y),

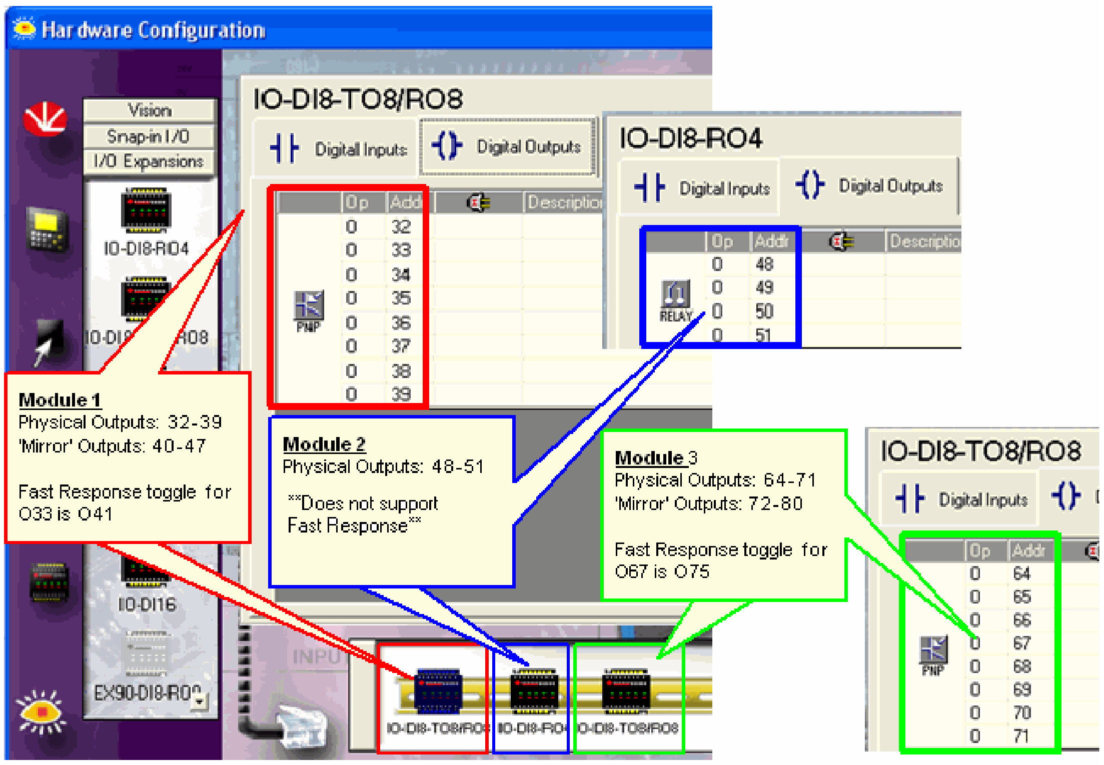
where X is the number representing the location of the module's location (0-7) and Y is the number of the input or output on that specific module (0-15).

To calculate the address of a 'toggle output', use:

(40 + x) + (16 + y)

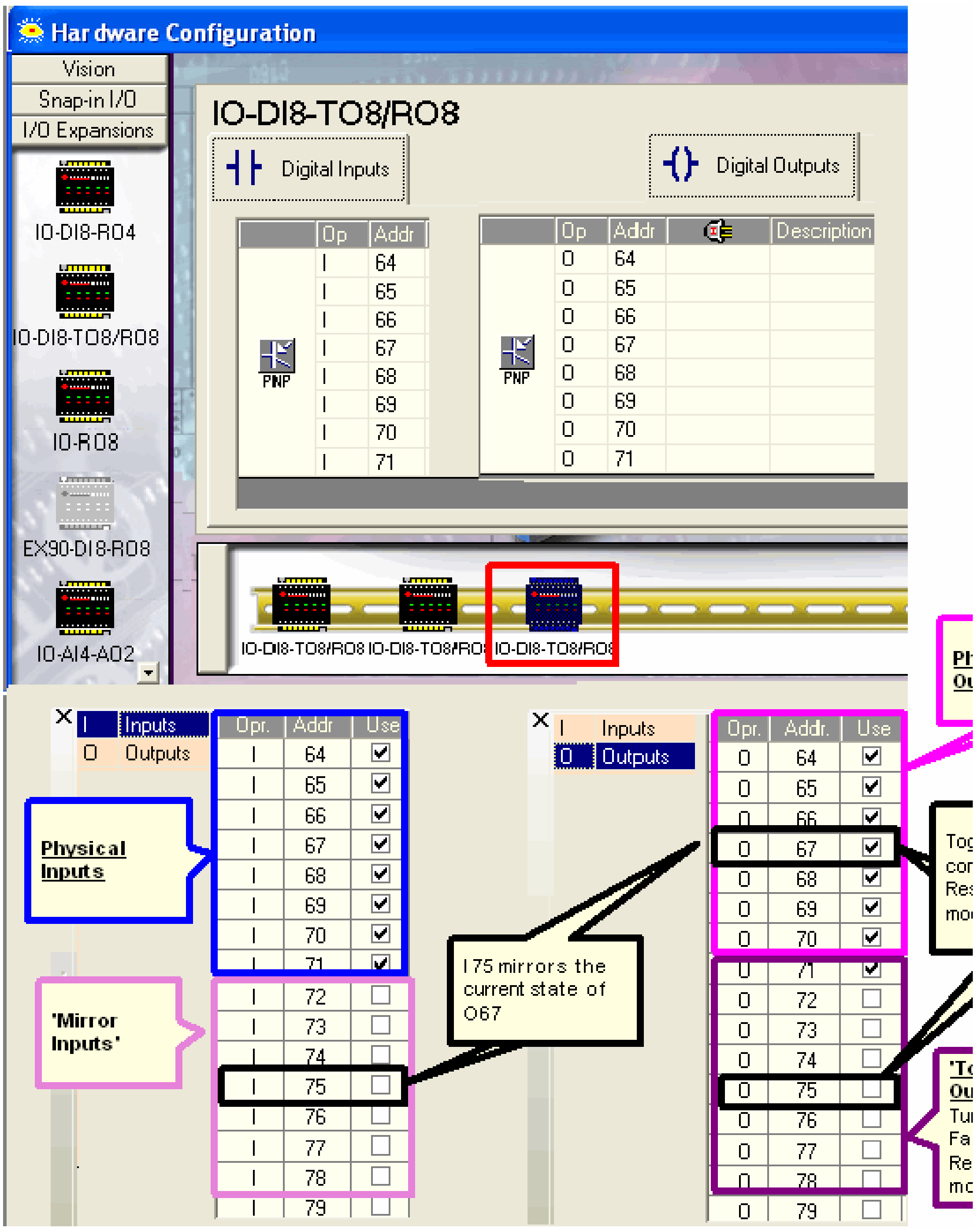
Example

- Output #4, located on module #3, will be addressed as O 84
  84 = (32 + 3) + (16 + 4)
  Its 'toggle output' will be O92
  92 = (40 + x) + (16 + 4)



About 'Mirror' Inputs

'Mirror' Inputs reflect the status of an output in Fast Response mode. In module I/O-DI8-TO8, the first 8 inputs in Hardware Configuration are related to the actual physical inputs. The second 8 inputs mirror the status of the module's physical outputs.
The figure below shows the third module in a project. If O67 enters Fast Response mode, the status of I75 will change to match the changing status of O67.



	Op	Addr
PNP	I	64
	I	65
	I	66
	I	67
	I	68
	I	69
	I	70
	I	71

	Op	Addr		Description
PNP	O	64		
	O	65		
	O	66		
	O	67		
	O	68		
	O	69		
	O	70		
	O	71		

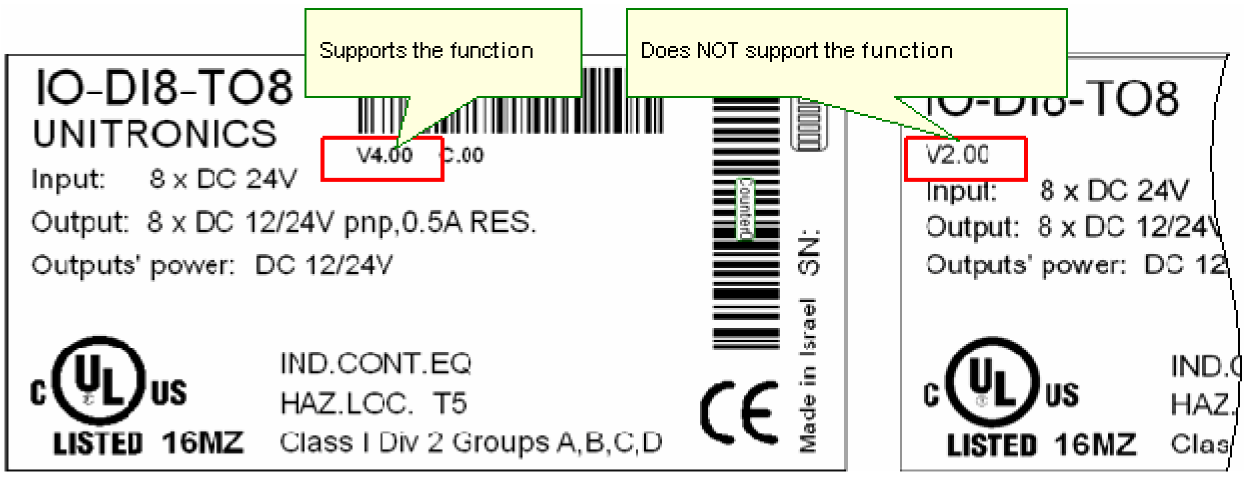
Opr.	Addr	Use
I	64	✓
I	65	✓
I	66	✓
I	67	✓
I	68	✓
I	69	✓
I	70	✓
I	71	✓
I	72	☐
I	73	☐
I	74	☐
I	75	☐
I	76	☐
I	77	☐
I	78	☐
I	79	☐

Opr.	Addr.	Use
O	64	✓
O	65	✓
O	66	✓
O	67	✓
O	68	✓
O	69	✓
O	70	✓
O	71	✓
O	72	☐
O	73	☐
O	74	☐
O	75	☐
O	76	☐
O	77	☐
O	78	☐
O	79	☐

Note • Older I/O Expansion Module models may not support Fast Response.

Checking the Model Version

The module bears a sticker which gives the version number. Versions previous to V4.00 do not support this function.


Supports the function
Does NOT support the function
IO-DI8-TO8
UNITRONICS
V4.00
Input: 8 x DC 24V
Output: 8 x DC 12/24V pnp,0.5A RES.
Outputs' power: DC 12/24V
IND.CONT.EQ
HAZ.LOC. T5
Class I Div 2 Groups A,B,C,D
Made in Israel SN:
V2.00

FB Operations

Fast Response operations are located on the FBs menu.

Configuration

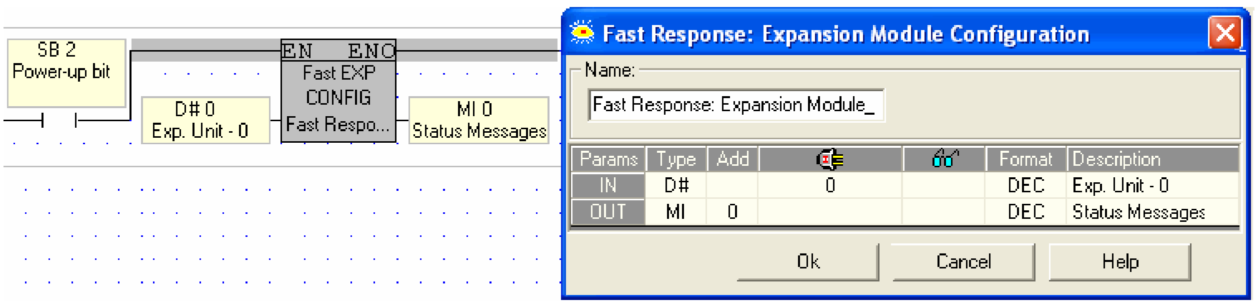
Set Mode

Set Counter

Scan

Fast Response: Configuration

Your program must activate the Configuration before running Fast Response operations. You can do this by activating it as a power-up task.

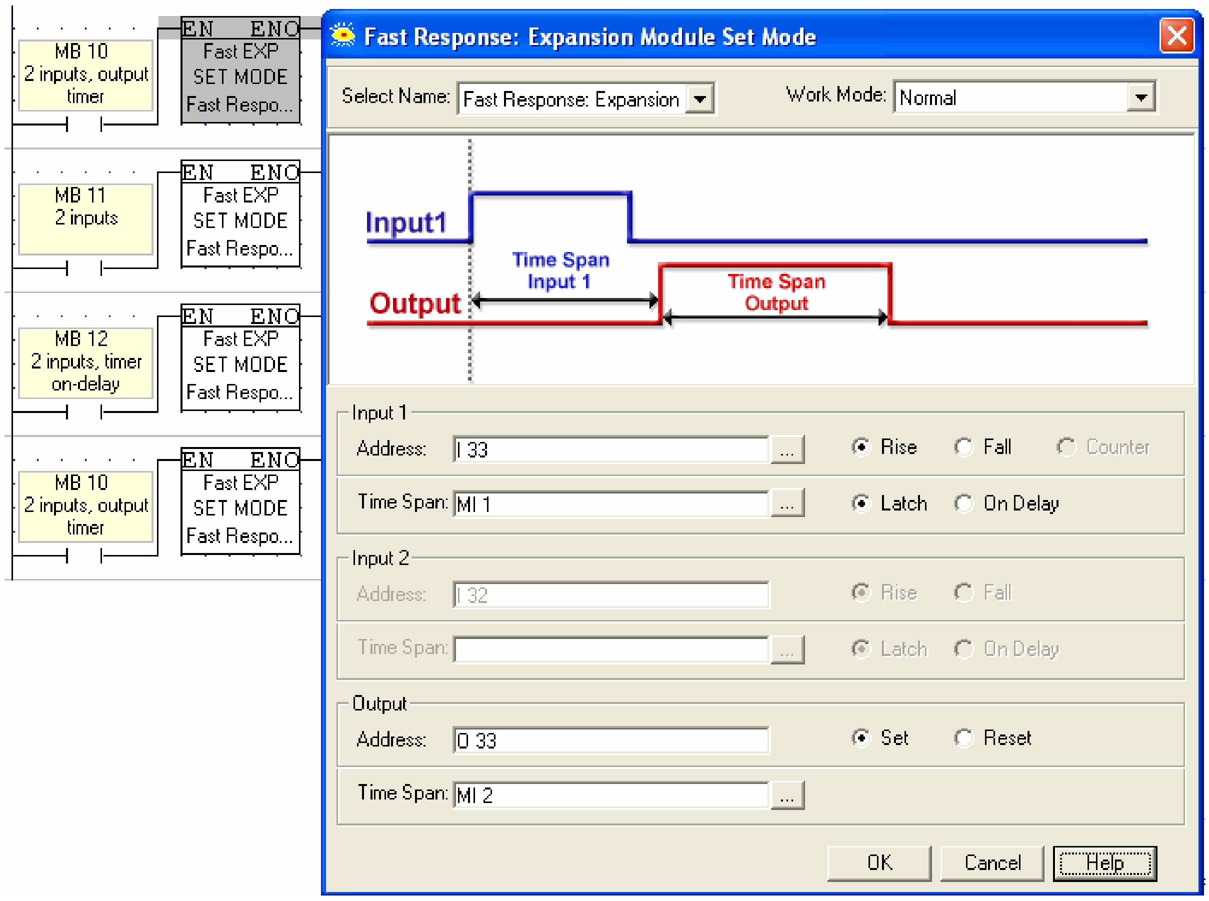


Params	Type	Add			Format	Description
IN	D#		0		DEC	Exp. Unit - 0
OUT	MI	0			DEC	Status Messages

Parameter	**Function**
Exp. Unit	This the DIN-rail position of the expansion module.
Status Messages	The value in the linked MI indicates the status of the function 0 - Function in progress. 1 - Complete without error. 2 - Buffer is full (impossible to send the command – please retry. Note that SB 91 (in M91, Next1 and Vision) is set when the buffer is full. 3 - The expansion module is processing another command – please retry. 6 - Communication Error (fatal) (expansion module does not exist).

Fast Response: Set Mode

Set Mode must follow the Fast Response Configuration. There are four modes available, as described in the section Mode Overview. The input and output options change according to the selected mode.



Select Name: Fast Response: Expansion
Work Mode: Normal

Input 1
Address: I 33 — [✓ Rise] Fall Counter
Time Span: MI 1 — [✓ Latch] On Delay

Input 2
Address: I 32 — [✓ Rise] Fall
Time Span: — [✓ Latch] On Delay

Output
Address: O 33 — [✓ Set] Reset
Time Span: MI 2

Set Mode Parameters

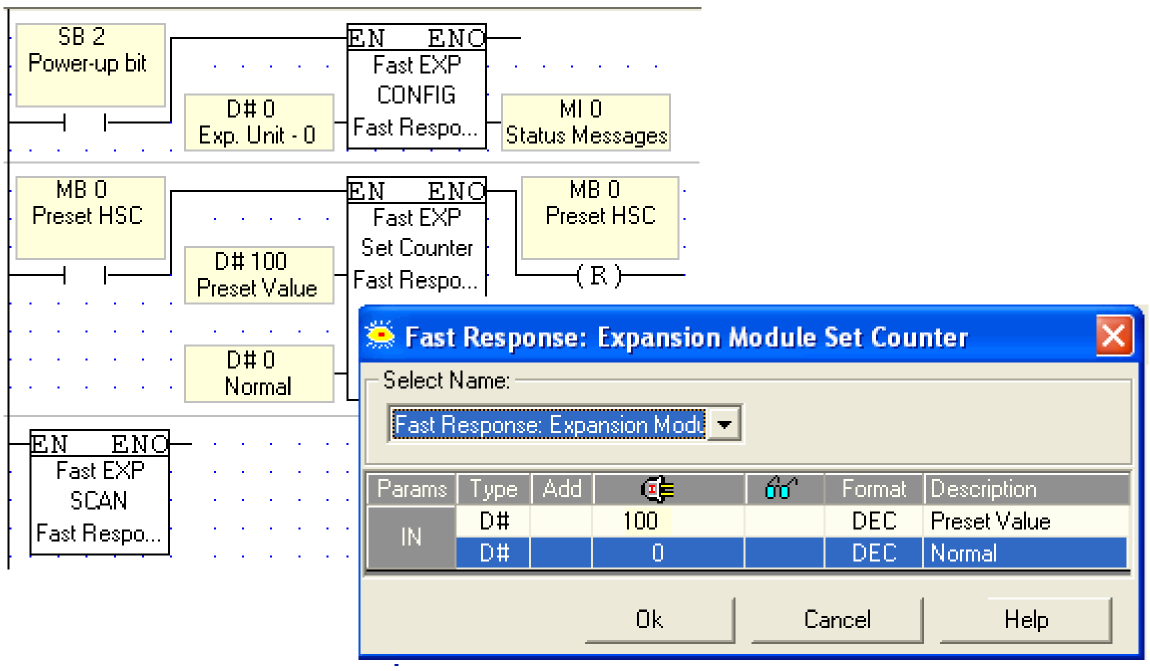
Type	Parameter	Function
Input	Address	The address of the actual input in the project's Hardware Configuration. When you select an input, the corresponding output is automatically selected: ex., selecting I0 automatically selects O0. Only odd-numbered addresses may be used.
	Rise/Fall	This is the event that causes the module to begin counting the input's Time Span
	Time Span	The value in the MI linked to Time Span provides a preset timer value. - Latch: once the time span is activated, time continues to elapse even if the input changes status. - On Delay: If the input changes status while the time span is being counted, the counting stops. When the input status changes again, the counter reset, and begins again from the preset value.
Output	Address	When you select an input, the corresponding output is automatically selected.
	Set/Reset	This sets the output's **Start State**.

		Set: Selecting Set means that the output's start state is OFF. After Input 1's Time span has elapsed, the output turns ON. Reset Selecting Reset means that the output's start state is ON. After Input 1's Time span has elapsed, the output turns OFF.
	Time Span	This is the amount of time the output will be ON or OFF

Notes •	Do not place Set Mode directly on the Ladder rail. Your program should activate Set Mode for a single program scan. If your application changes modes, activate Set Mode for a single scan at each change.
•	In Two Input modes; note that you can only assign odd addresses to Input 1.
•	When an input is selected, the function controls the corresponding output. For example, if I35 is selected, O35 will be controlled by the function.
•	In Two Input modes, when the first input address is selected, the function automatically selects the **previous** address for the second input.

Fast Response: Set Counter

Set Counter enables you to store a value into the I/O Expansion Module's high speed counter. You can also initialize the counter by storing the value zero.



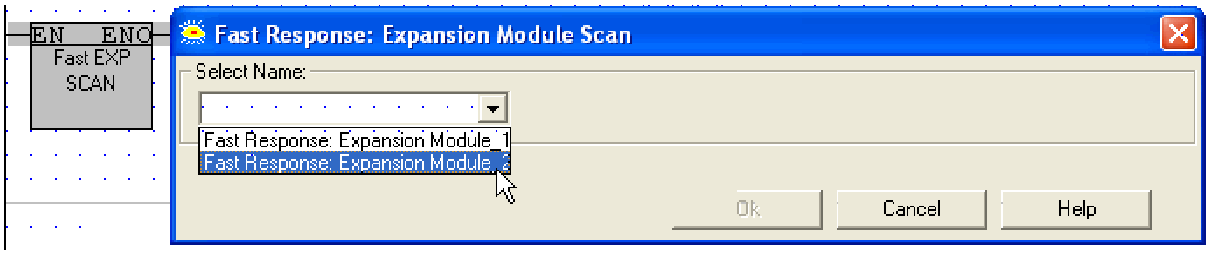
Select Name: Fast Response: Expansion Modu

Params	Type	Add			Format	Description
IN	D#		100		DEC	Preset Value
	D#		0		DEC	Normal

Notes • Do not place Set Counter directly on the Ladder rail. Your program should activate Set Counter for a single program scan, each time you wish to set the counter.

Fast Response: Scan

A Scan FB must be included for every Fast Response Configuration in the application. When an application contains more than 1 Configuration in the Ladder, the Scan FB displays the Select Name drop-down list, enabling you to link the desired Configuration.



Fast Response: Expansion Module_1
Fast Response: Expansion Module_2

Note • The Scan FB must be on the left-hand ladder rail.

BAS (Building Automation Systems)

BAS, CSI Overview

The BAS operations enable Vision controllers to exchange data with CSI building environment cards. BAS operations are located on the FBs toolbar.

To use BAS, you must initialise the PLC COM port to 9600, 8n1.

BAS Operations

Configuration

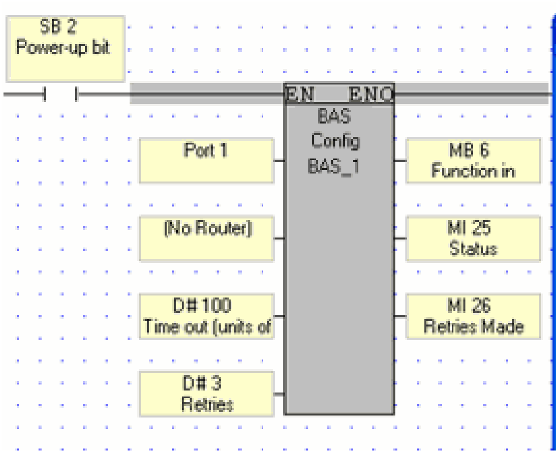
Scan

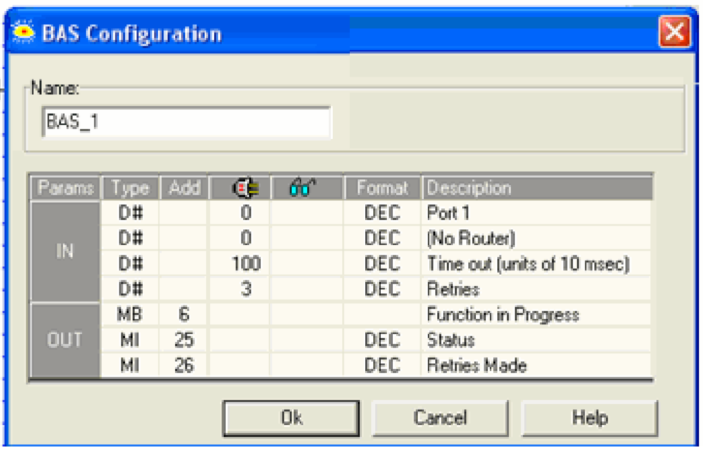
Open Session

Read, Write Inputs: Digital or Analog I/Os

Configuration

The BAS Configuration must be included in the application. In addition, Scan is required in order to enable the PLC application to receive BAS messages; the Open Session operation must also be included in the application.





Name: BAS_1

Params	Type	Add			Format	Description
IN	D#		0		DEC	Port 1
	D#		0		DEC	(No Router)
	D#		100		DEC	Time out (units of 10 msec)
	D#		3		DEC	Retries
OUT	MB	6				Function in Progress
	MI	25			DEC	Status
	MI	26			DEC	Retries Made

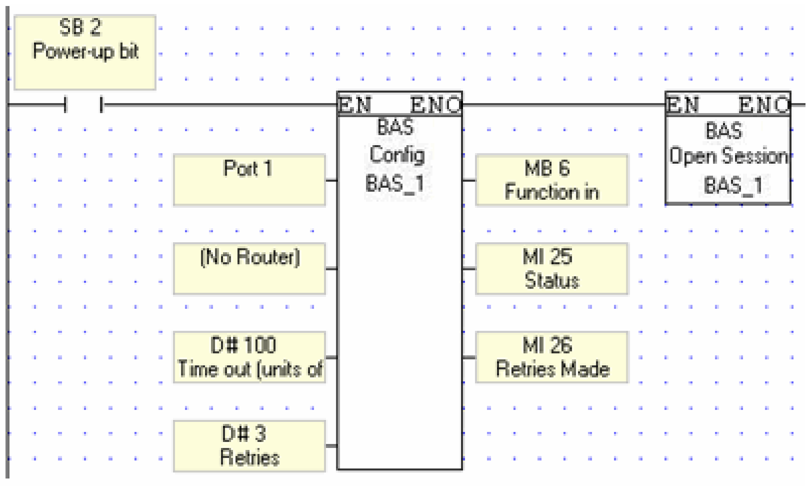
Configuration Parameters

Parameter	Type	Function
Port #	Constant	Enter the number of the PLC serial port that connects the PLC to the BAS network.
Router ID	Constant	If the BAS card is accessed via a router, select the ID number.
Time Out	MI or Constant	Amount of time the PLC will wait for a reply, units of 10 mS.
Retries	MI or Constant	Number of times the PLC will attempt to send a message.
Function in Progress	MB	Turns ON when the PLC sends a BAS data request, remains ON either until the Time Out is exceeded or until a message is received in answer to the data request. Use this MB as a condition to send a message.
Retries Made	MI or Constant	Number of times the PLC has attempted to send a message.
Status	MI	The value of the linked MI indicates messages as follows: Value Message 1 Operation completed successfully 2 (Transmit) Illegal card ID (0-255) 3 (Transmit) Illegal point value 4 (Transmit) Illegal bit offset 5 (Transmit)-Serial port busy 6 (Receive) Illegal length The actual length of the message does not match the length given in the first message byte 7 (Receive) Illegal card ID The ID received does not match the ID in the data request message 8 (Receive) Illegal Router

		The message was received from a router that is not the router named in the data request message 9 Illegal Command The command received does not match the command in the data request message 10 (Receive) Illegal point value The point received does not match the point in the data request message 11 (Receive) TimeOut An answer was not received before the timeout set in the BAS Configuration. 12 (Receive) Checksum error 13 (Receive) Illegal offset for IO type

Open Session

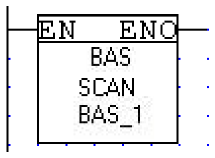
In order to run BAS operations, you must include Open Session in the application; it may be activated as a power-up task.



Scan

To enable the controller to receive messages, place an BAS Scan FB in your application and link it to a Configuration. When activated, this causes the controller to scan the Com port for incoming BAS messages.

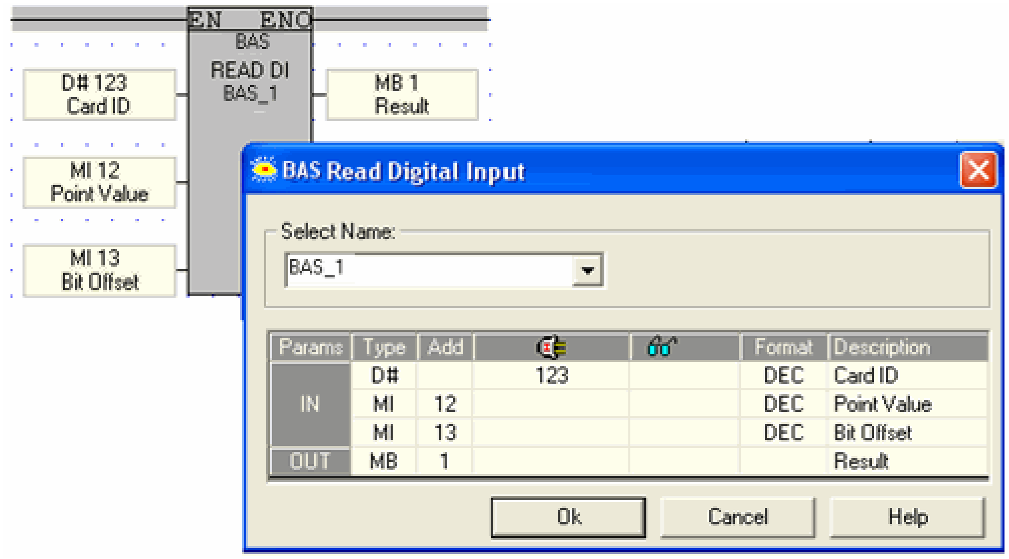
Before you can receive a BAS message, you must initialize a COM port.



Read, Write Inputs: Digital or Analog I/Os

You can read from and write to digital or analog I/Os

Digital I/Os

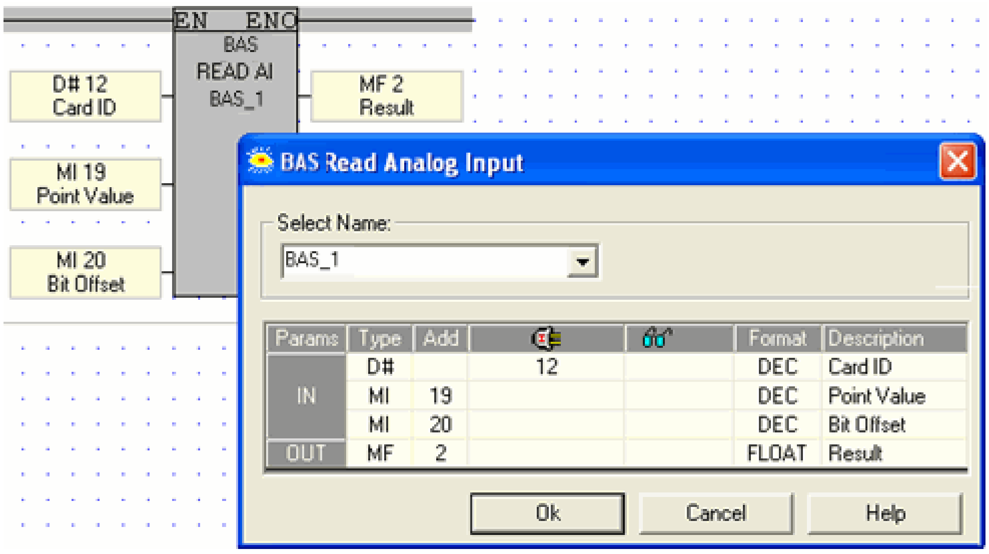


Select Name: BAS_1

Params	Type	Add			Format	Description
IN	D#		123		DEC	Card ID
	MI	12			DEC	Point Value
	MI	13			DEC	Bit Offset
OUT	MB	1				Result

Parameter	Type	Function
Card ID	MI or Constant	Enter the ID number of the card to be read from/written to.
Point Value	MI or Constant	Card Point to be read from/written to.
Bit offset	MI or Constant	Enter the bit offset value.
Result	MB	Read: Writes the current status of the I/O to the selected MB. Write: Writes the current status of the MB to the selected I/O.

Analog I/Os



Select Name: BAS_1

Params	Type	Add			Format	Description
IN	D#		12		DEC	Card ID
	MI	19			DEC	Point Value
	MI	20			DEC	Bit Offset
OUT	MF	2			FLOAT	Result

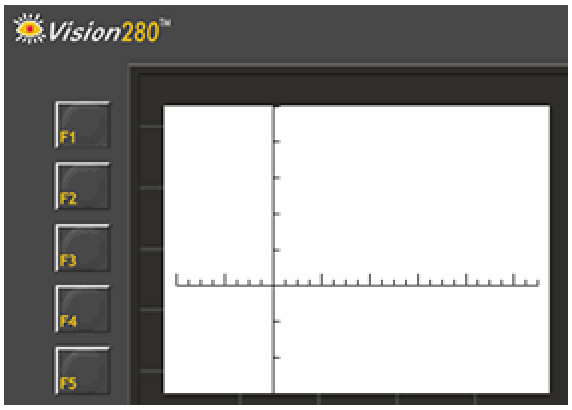
Parameter	**Type**	**Function**
Card ID	MI or Constant	Enter the ID number of the card to be read from/written to.
Point Value	MI or Constant	Card Point to be read from/written to.
Bit offset	MI or Constant	Enter the bit offset value.
Result	MF	Read: Writes the current value of the I/O to the selected MF. Write: Writes the current value of the MF to the selected I/O.

Draw Axis

Draw Axis Overview

Use this function to place x and y axes, including ticks, on the Vision screen in response to Ladder conditions.

The axes may be used to provide a background for bar graphs, or in conjunction with the Trends function block.



Draw Axis operations are located on the FBs toolbar.

FB Operations

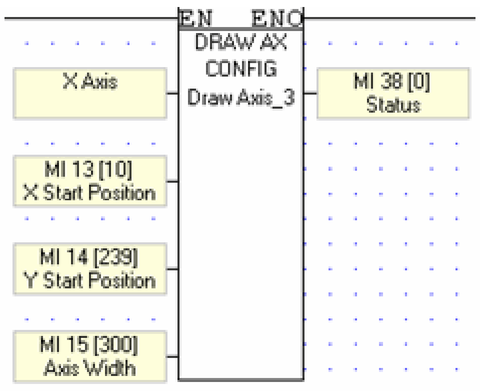
Configuration

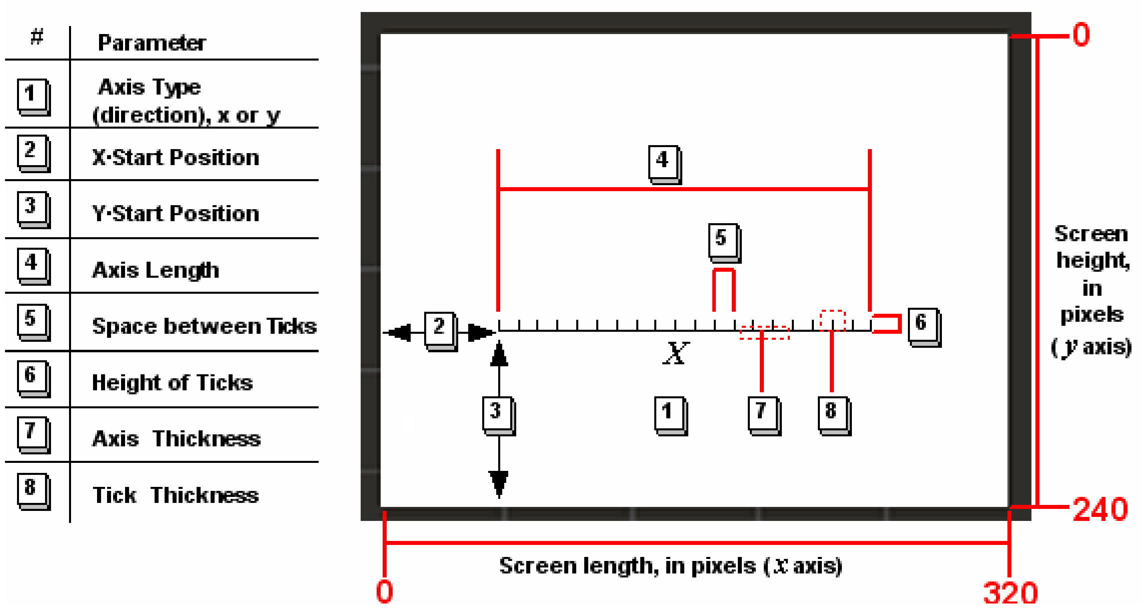
Draw

Clear

Configuration

The Draw Axis Configuration sets the parameters the controller uses to draw the axis on the LCD. Each Draw Axis operation is linked to a Configuration.



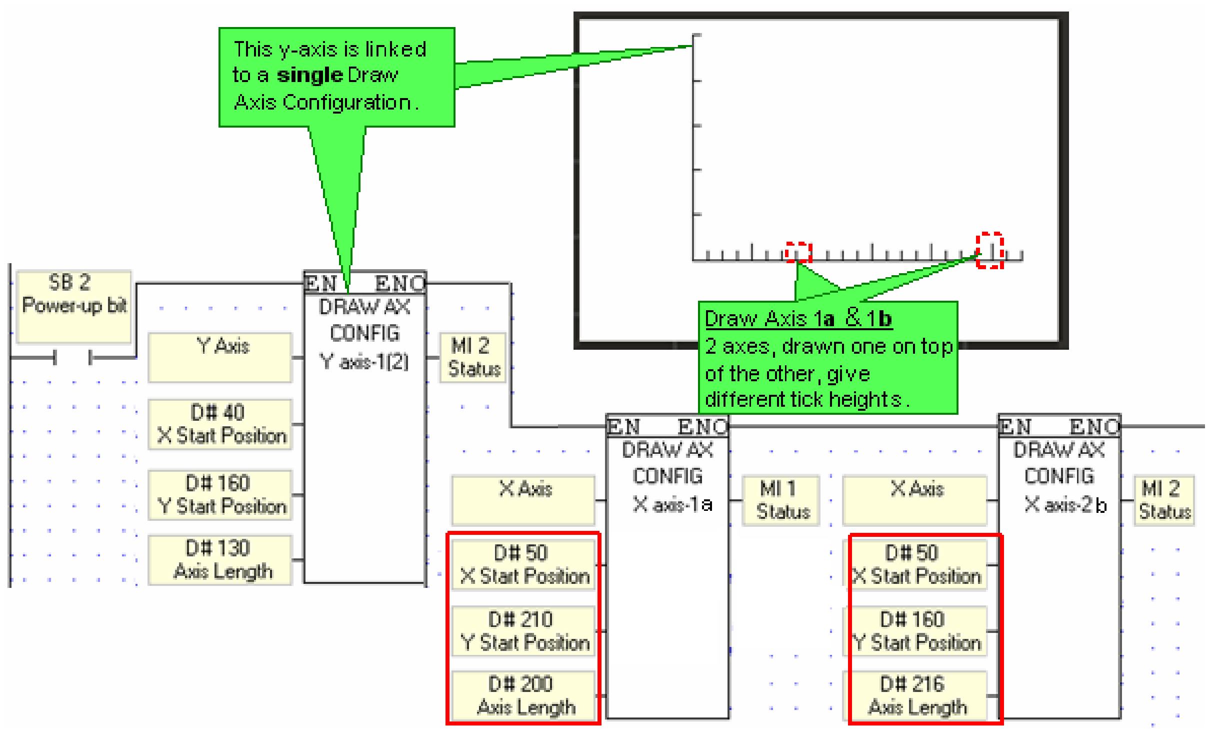


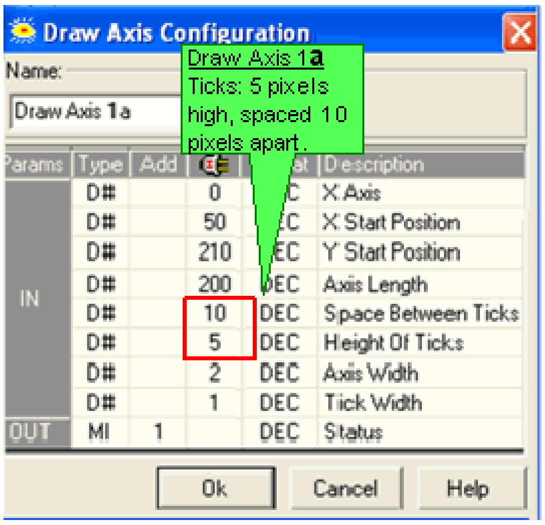
Parameter	Type	Function
Axis Type	Constant	This sets the axis direction. Select between: X axis (horizontal). Y axis (vertical).
X Start Position	MI,ML, DW, or Constant	x-Origin of the axis, in pixels.
Y Start Position	MI,ML, DW, or Constant	y-Origin of the axis, in pixels.
Axis Length	MI,ML, DW, or	Length of the axis, in pixels.

	Constant	
Space Between Ticks	MI,ML, DW, or Constant	Distance between ticks, in pixels.
Height Of Ticks	MI,ML, DW, or Constant	Height of ticks, in pixels.
Axis Thickness	MI,ML, DW, or Constant	Thickness of the main axis line, in pixels.
Tick Thickness	MI,ML, DW, or Constant	Thickness of ticks, in pixels.
Status	MI	If the axis or ticks are not drawn when the Draw operation is called, check the value of the Status MI. The first 4 (LSB) bits of the MI act as a bitmap to indicate the messages listed below.Bit Message 0 PLC in Info Mode, Axis cannot be drawn 1 PLC in Info Mode, Axis cannot be cleared (erased). 2 The main axis line cannot be drawn because: - the length of the line exceeds the screen's dimensions. - the coordinates of the line are not within the screen. - both of the above. 3 The ticks cannot be drawn because: - the length of the ticks exceeds the screen's dimensions. - the ticks are not within the screen.

Each axis, whether x or y, that is drawn on screen requires a separate Draw Axis Configuration. In addition, note that in order to obtain an axis with different tick heights, you must superimpose one axis on top of another.

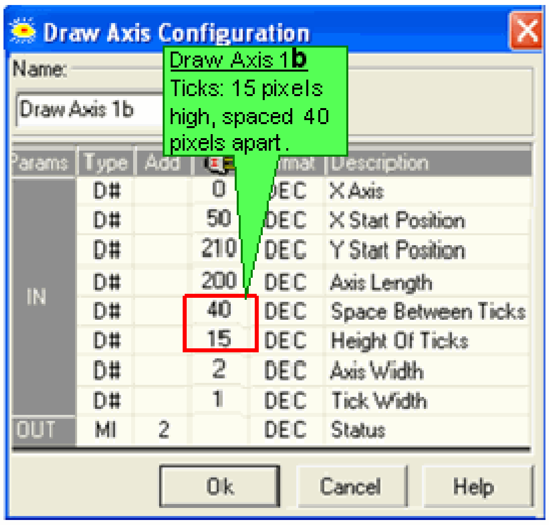
The example below shows a horizontal axis that is composed of 2 separate Draw Axis Configurations. Note that the parameters supplying coordinates are identical; the tick height and spacing are different.





Name: Draw Axis 1a

Params	Type	Add		Format	Description
IN	D#		0	DEC	X Axis
	D#		50	DEC	X Start Position
	D#		210	DEC	Y Start Position
	D#		200	DEC	Axis Length
	D#		10	DEC	Space Between Ticks
	D#		5	DEC	Height Of Ticks
	D#		2	DEC	Axis Width
	D#		1	DEC	Tick Width
OUT	MI	1		DEC	Status



Name: Draw Axis 1b

Params	Type	Add		Format	Description
IN	D#		0	DEC	X Axis
	D#		50	DEC	X Start Position
	D#		210	DEC	Y Start Position
	D#		200	DEC	Axis Length
	D#		40	DEC	Space Between Ticks
	D#		15	DEC	Height Of Ticks
	D#		2	DEC	Axis Width
	D#		1	DEC	Tick Width
OUT	MI	2		DEC	Status

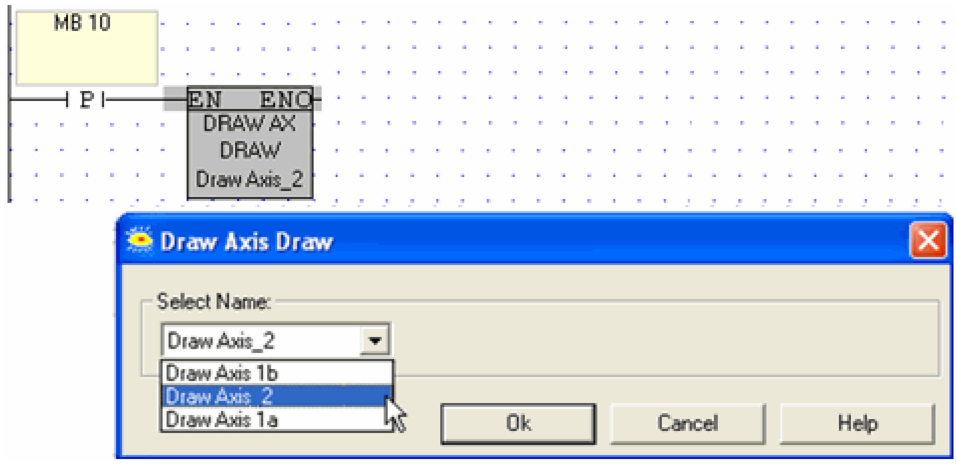
Draw

To display the axis on screen:

1. Place a Draw operation in the Ladder application.
2. Link it to the desired Configuration.

When Draw is activated, generally by a positive transitional contact, the axis will appear on screen. If the axis does not appear, check the Status MI in the Draw Axis Configuration.

The axis remains on screen until it is cleared by a Clear operation.

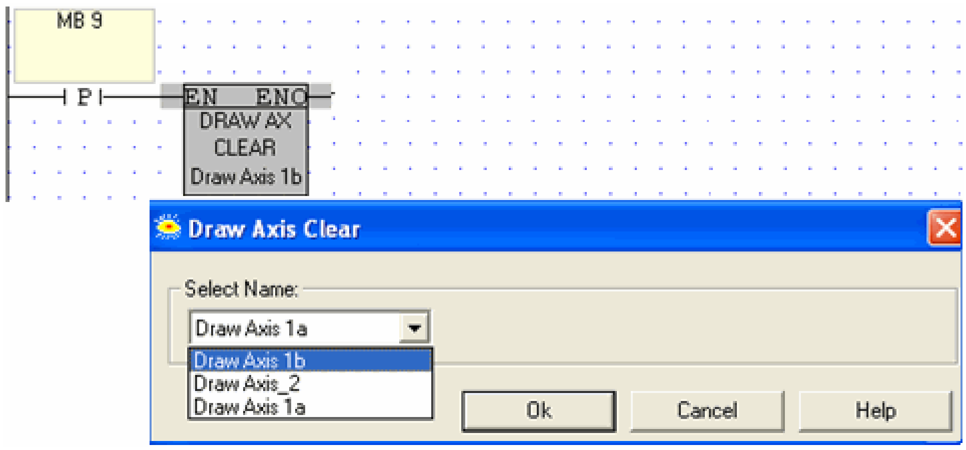


Clear

To clear the axis from the screen:

1. Place a Clear operation in the Ladder application.
2. Link it to the desired Configuration.

When Clear is activated, generally by a positive transitional contact, the axis disappears from the screen.



Note • The Draw Axis Clear clears the last axis drawn.

This means that if the application:

Draws an axis,

Redraws the same axis, but after changing the Configuration's parameters, changing the location/appearance of the axis,

Runs Clear Draw Axis

The result is that only the axis drawn in Step 2 will be cleared.

This means that you must run Clear Draw Axis **before** redrawing an axis.